Theoretical Computer Science

Unit 3: Grammars

Faculty Name : Ms. Namita Pulgam

Index –

D Y PATIL DEEMED TO BE UNIVERSITY
RAMRAO ADIK INSTITUTE OF TECHNOLOGY
NAVI MUMBAI

Lecture No 36:

Conversion of CFG to GNF (Part-3) & Revision

Example 4:

Q. Convert following grammar to CNF and GNF.

$$E \rightarrow E+E \mid E*E \mid (E) \mid id$$

Consider "id" as a single terminal/symbol.

Solution :

- **Step 1 : Given CFG is not having any Unit Productions or Null Productions. But it is not in CNF so first converting this to CNF.**
- **Step 2 : Lets add new productions as shown :**

  $R_1 \rightarrow +$

  $R_2 \rightarrow *$

  $R_3 \rightarrow ($

  $R_4 \rightarrow)$

- **Step 3 : Updated grammar is as follows :**

  $E \rightarrow E\, R_1\, E \mid E\, R_2 E \mid R_3 E\, R_4 \mid id$

Example 4: (Cont..)

- **Step 4 : Identify non CNF productions**

  $E \rightarrow E R_1 E \mid E R_2 E \mid R_3 E R_4$

- **Step 5 : In productions only two non terminals are allowed so break them as shown :**

  $E \rightarrow ER_5$

  $R_5 \rightarrow R_1 E$

  $E \rightarrow ER_6$

  $R_6 \rightarrow R_2 E$

  $E \rightarrow R_3 R_7$

  $R_7 \rightarrow ER_4$

- **Step 6 : Thus given Grammar is in CNF having following productions :**

  $E \rightarrow ER_5 \mid ER_6 \mid R_3 R_7 \mid id$

  $R_1 \rightarrow +$

  $R_2 \rightarrow *$

  $R_3 \rightarrow ($

  $R_4 \rightarrow)$

  $R_5 \rightarrow R_1 E$

  $R_6 \rightarrow R_2 E$

  $R_7 \rightarrow E R_4$

Example 4: (Cont..)

- **Step 7 : Check order of variables present in CNF.**

  - E is lowest variable while R_7 is higher variable. Hence the production $R_7 \rightarrow E\, R_4$ is in **non GNF.**

  - Using substitution convert production into required form :

$$R_7 \rightarrow \mathbf{ER_5}R_4 \mid \mathbf{ER_6}R_4 \mid \mathbf{R_3R_7}R_4 \mid \mathbf{id}R_4 \qquad (\text{Replacing E by } ER_5 \mid ER_6 \mid R_3R_7 \mid id)$$

- **Step 8 : Removal of Left Recursive Production :**

  - After conversion we get left recursive production i.e. $E \rightarrow ER_5 \mid ER_6 \mid R_3R_7 \mid id$

  - Comparing this with $A \rightarrow A\alpha_1 \mid A\alpha_2 \mid A\alpha_3 \mid \ldots\ldots\ldots\ldots \mid \beta_1 \mid \beta_2 \mid \beta_3 \ldots\ldots\ldots$

  - From above production we get : $\alpha_1 = R_5$

$$\alpha_2 = R_6$$
$$\beta_1 = R_3R_7$$
$$\beta_2 = id$$

  - Let B be new variable to be introduced then above productions can be replaced by :

$$E \rightarrow R_3R_7 \mid id \mid R_3R_7B \mid idB$$
$$B \rightarrow R_5 \mid R_6 \mid R_5B \mid R_6B$$

Example 4: (Cont..)

- **Step 9 : Perform substitution**

  - In production of E , E $\rightarrow$ R_3R_7 | id | R_3R_7B | idB , replace R_3 by (:

    E $\rightarrow$ (R_7 | id | (R_7B | idB

  - Consider productions of R_5 and R_6 replace R_1 and R_2 by + and * respectively :

    $R_5 \rightarrow +E$ (Since $R_5 \rightarrow R_1E$ replace by $R_1 \rightarrow +$)

    $R_6 \rightarrow *E$ (Since $R_6 \rightarrow R_2E$ replace by $R_2 \rightarrow *$)

  - In production of B, B $\rightarrow$ R_5 | R_6 | R_5B | R_6B replace R_5 and R_6 by its production, :

    B $\rightarrow$ **+E** | ***E** | **+E** B | ***E**B

  - Production of R_7 is $R_7 \rightarrow ER_5R_4$ | ER_6R_4 | $R_3R_7R_4$ | idR_4 , replace E by its production i.e. E $\rightarrow$ (R_7 | id | (R_7B | idB and R_3 by its production $R_3 \rightarrow$ (and we get,

    $R_7 \rightarrow$ ($R_7R_5R_4$ | idR_5R_4 | ($R_7BR_5R_4$ | $idBR_5R_4$ | ($R_7R_6R_4$ | idR_6R_4 | ($R_7BR_6R_4$ | $idBR_6R_4$ | (R_7R_4 | idR_4

Example 4: (Cont..)

- **Step 10 : Final grammar in GNF :**

$E \rightarrow (R_7 \mid id \mid (R_7B \mid idB$

$B \rightarrow +E \mid *E \mid +E\,B \mid *EB$

$R_1 \rightarrow +$

$R_2 \rightarrow *$

$R_3 \rightarrow ($

$R_4 \rightarrow)$

$R_5 \rightarrow +E$

$R_6 \rightarrow *E$

$R_7 \rightarrow (R_7R_5R_4 \mid idR_5R_4 \mid (R_7BR_5R_4 \mid idBR_5R_4 \mid (R_7R_6R_4 \mid$
$idR_6R_4 \mid (R_7BR_6R_4 \mid idBR_6R_4 \mid (R_7R_4 \mid idR_4$

Revision Quiz

1. Given Grammar G:

S → aA

A → a| A

B →B

The number of productions to be removed immediately as Unit productions:___

a. 0

b. 1

c. 2

d. 3

Answer: c

2. Identify Unit Production___________

a. X → YZ

b. X → 1

c. X → Y

d. XY →ZM

Answer: c

D Y PATIL DEEMED TO BE UNIVERSITY — RAMRAO ADIK — INSTITUTE OF TECHNOLOGY NAVI MUMBAI

Revision Quiz continued..

3. Simplify following Grammar

S → aSb| a |aAB

A → aA| c

B → bB |dB

a. S → aSb| a |aAB

A → aA| c

b. S → aSb| a |aAB

A → aA| c

B → bB |dB

c. S →aSb| a

d. S → a

Answer: c

Revision Quiz continued..

4. How many components in Formal definition of Grammar?

4

5

2

3

Answer : a

5. G = { (S, A), (a, b), P, S }

where P: S $\rightarrow$ aAS | a

A $\rightarrow$ SbA | SS | ba

It generates which string?

a. aabaa

b. ababab

c. aabbaa

d. abba

Answer : c

Revision Quiz continued..

6. _________ is Type 0 grammar.

a. Regular Grammar

b. Context Sensitive Grammar

c. Context Free Grammar

d. Unrestricted Grammar

Answer :d

7. What is the highest type number which can be applied to the following grammar ?

$S \rightarrow Aa, A \rightarrow Ba, B \rightarrow abc$

a. Type 0

b. Type 1

c. Type 2

d. Type 3

Answer : c

Revision Quiz continued..

8. The geometrical representation of a derivation is called as ______

a. Parse Tree

b. Derivation Tree

c. Syntax Tree

d. All of the above mentioned

Answer : d

9. The format: A → aB refers to which of the following?

a. Chomsky Normal Form

b. Greibach Normal Form

c. Backus Naur Form

d. None of the mentioned

Answer : b

10. A grammar that produces more than one parse tree for some sentence is called as____

a. Ambiguous

b. Unambiguous

c. Regular

d. All of these

Answer : a

Revision Quiz continued..

11. Every grammar in Chomsky Normal Form is:

a. regular

b. context sensitive

c. context free

d. all of the mentioned

Answer : c

12. A → aA| a| b. The number of steps to form aab:_____

a. 2

b. 3

c. 4

d. 5

Answer : b

13. Suppose A → xBz | y and B → y, then the simplified grammar would be:_______

a. A → y

b. A → xBx | xyz

c. A → xBz | B | y

d. none of the mentioned

Answer : d

Revision Quiz continued..

14. Application of CFG______

a. construction of compilers

b. parsing the program by constructing syntax tree

c. describing arithmetic expressions

d. all of the above mentioned

Answer : d

15. G = { (S, X), (a, b), P, S } where P : S → aSX | b, X → Xb| a. To derive aababa how many parse trees can be drawn?

a. 2

b. 3

c. 1

d. 0

Answer : c

Thank You

Theoretical Computer Science

Unit 4: Pushdown Automata

Faculty Name : Ms. Namita Pulgam

Lecture 37

Pushdown Automata

Introduction

- Finite automata (FA) cannot remember information.
- A string of the form $\mathbf{a^m b^m}$ can't be handled by finite automata, as it is required to remember number of a's to check equal b's.
- Just as a DFA is a way to implement a regular expression, a Pushdown Automata (PDA) is a way to implement a **context free grammar**

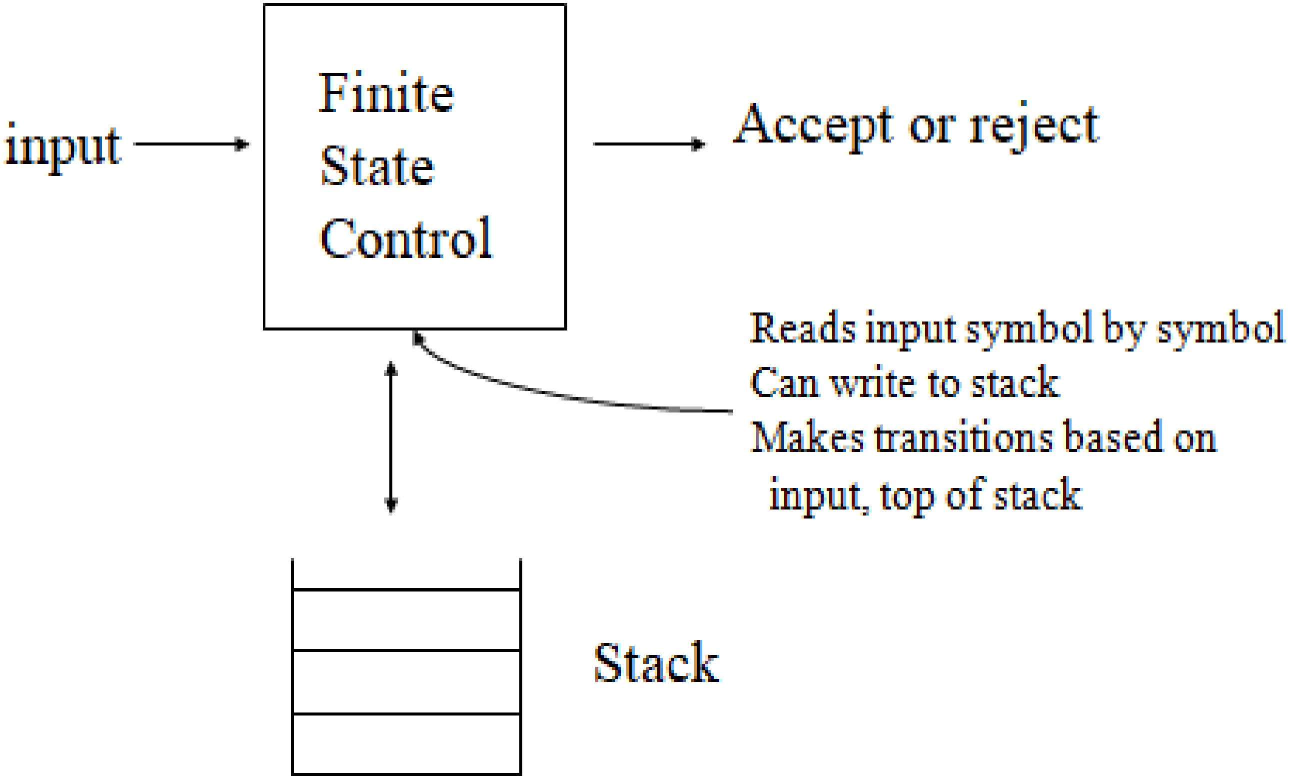
  - PDA equivalent in power to a CFG
- Essentially identical to a regular automata except for the addition of a stack

  - Stack is of infinite size

  - Stack allows us to recognize some of the non-regular languages

D Y PATIL DEEMED TO BE UNIVERSITY
RAMRAO ADIK INSTITUTE OF TECHNOLOGY
NAVI MUMBAI

Introduction (Cont..)



Formal Definition of PDA

$P = (Q, \sum, \Gamma, \delta, q_0, z_0, F)$

Q = finite set of **states**, like the finite automaton

$\sum$ = finite set of **input symbols**, the alphabet

Γ = finite **stack alphabet**, components we are allowed to push on the stack

δ = finite set of **transitions**

q_0 = start state

z_0 = start symbol.

-Initially, the PDA's stack consists of one instance of this start symbol and nothing else. We can use it to indicate the bottom of the stack.

F = Set of **final accepting states**.

Implementation of PDA

- **In one transition the PDA may do the following :**
  - Consume the input symbol. If ε is the input symbol, then no input is consumed.
  - Go to a new state, which may be the same as the previous state.
  - Replace the symbol at the top of the stack by any string as :
    - Replace with a new string (push operation)
    - Replace with multiple symbols (multiple pushes)
    - Replace with ε symbol (pop operation)
    - The string might be the same as the current stack top (does nothing)

Operations in PDA

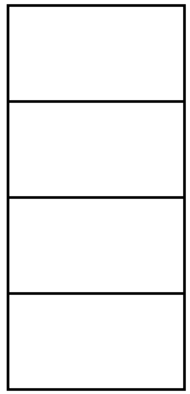
1. **Push :** This operation pushes a input symbol on stack and modifies a stack top.

2. **Pop :** This operation pops symbol on stack top and modifies a stack top. **(ε is pushed on stack)**

3. **No-operation :** This operation does not modify stack.

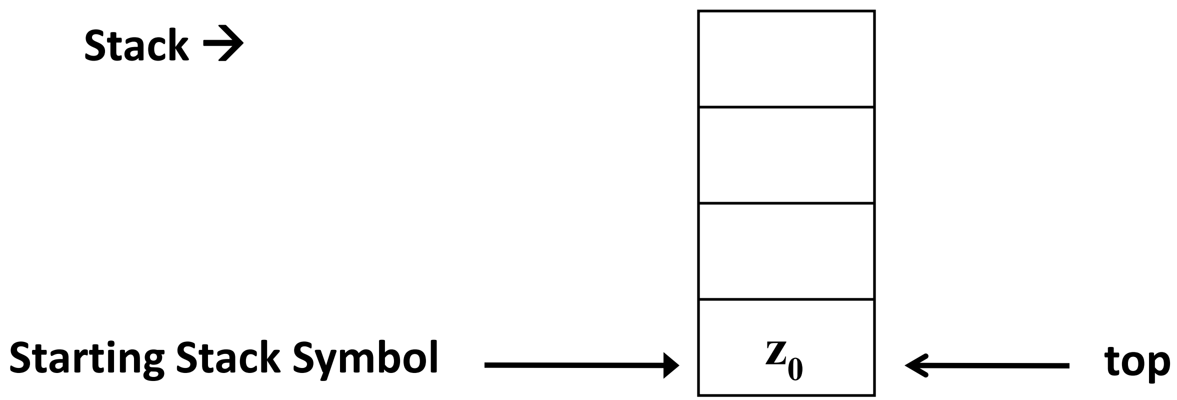
Initial Stack

Stack →



At Initial State (Appears at time 0) :

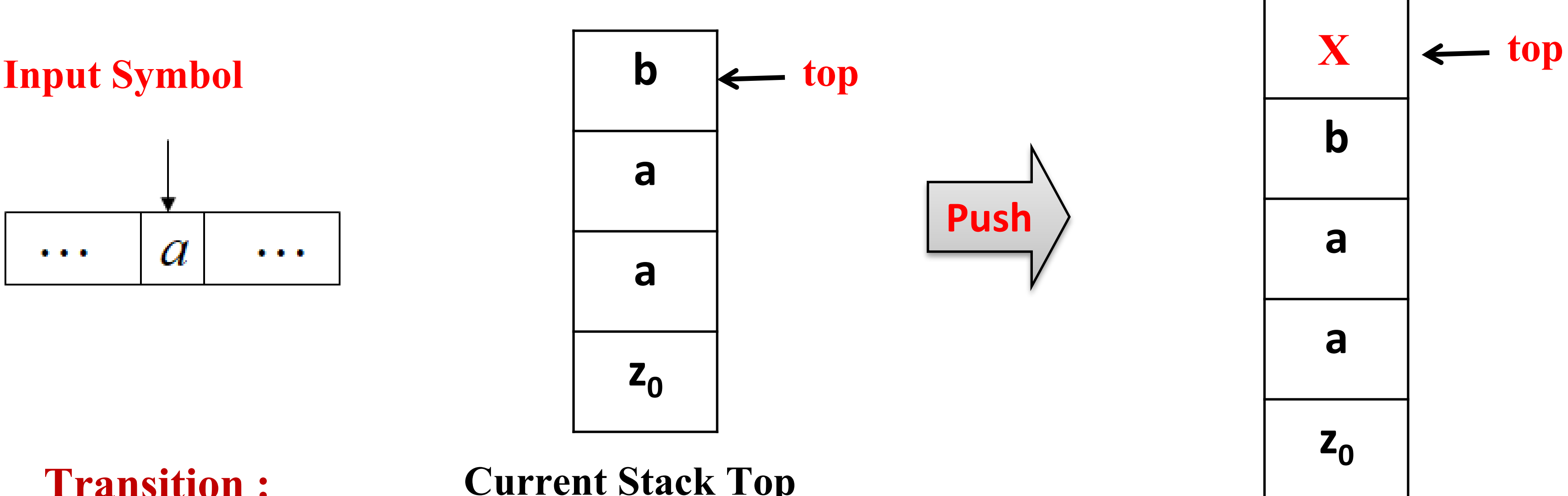
Stack →



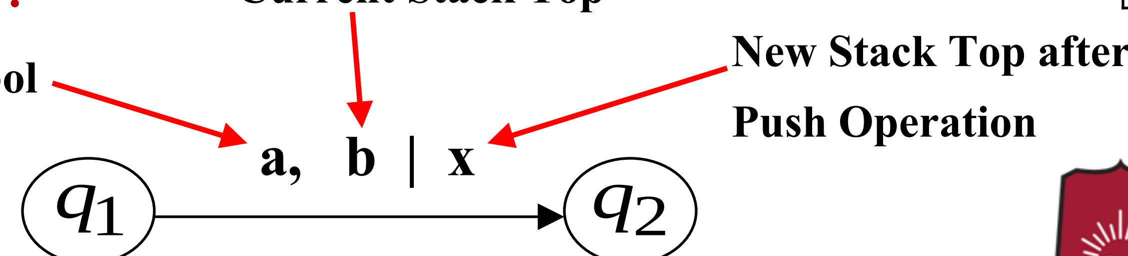
Push Operation

If input '**a**' is given to state 'q_1' and current stack symbol is '**b**' then state changes to 'q_2' and **x** is pushed on stack.

Working with Stack :



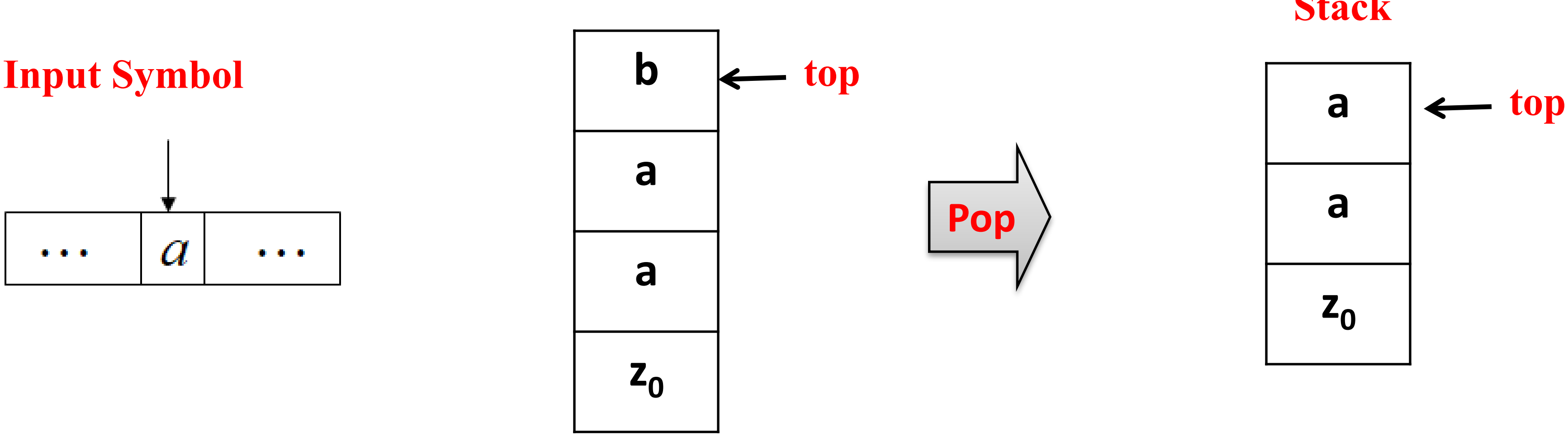
Transition :



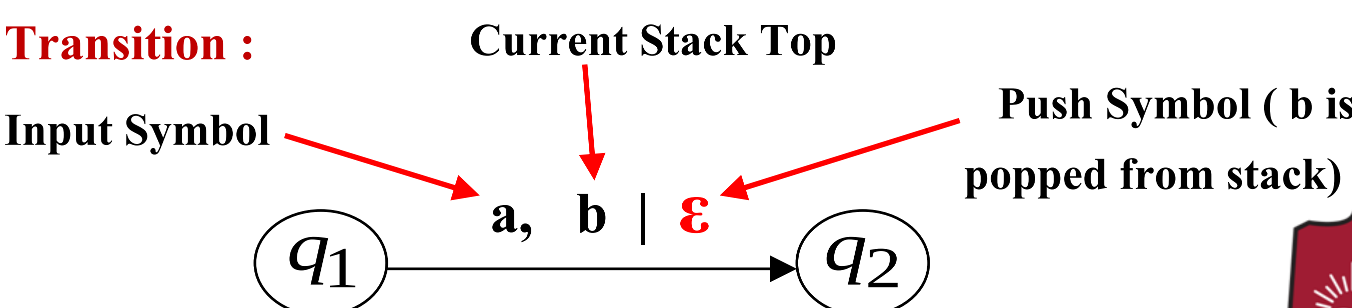
Pop Operation

If input '**a**' is given to state 'q_1' and current stack symbol is '**b**' then state changes to 'q_2' and **b** is popped from the stack.

Working with Stack :



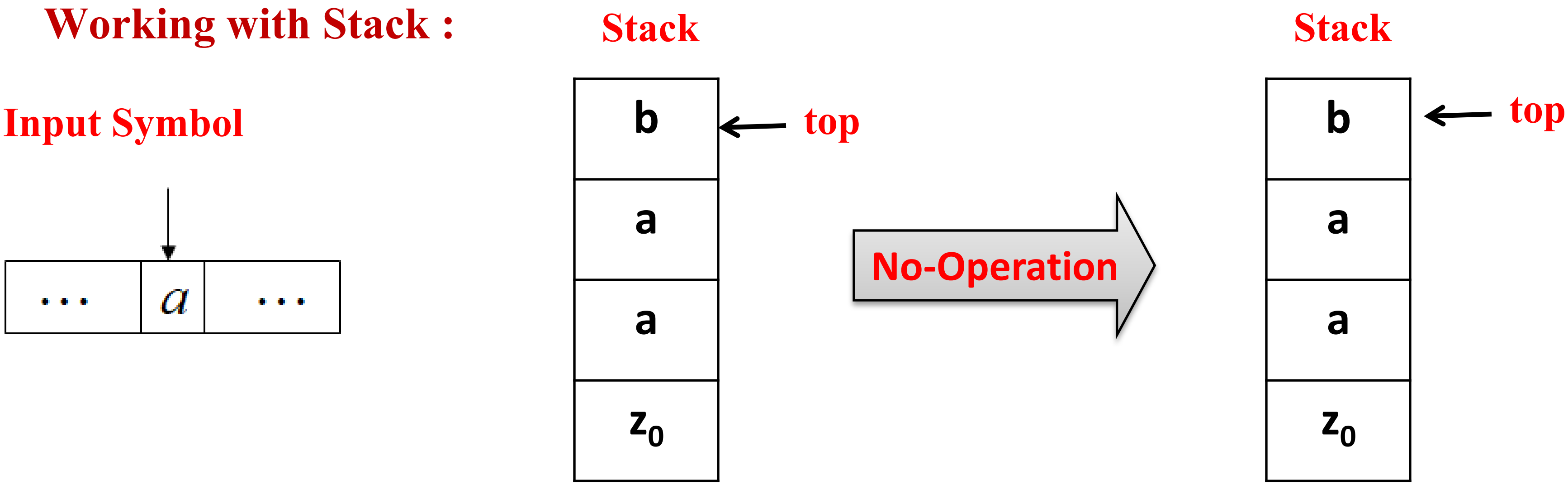
Transition :



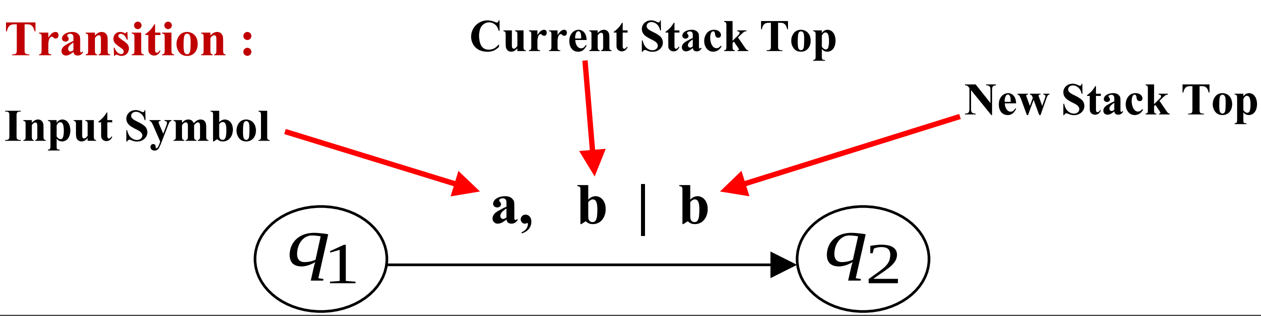
No Operation

If input '**a**' is given to state 'q_1' and current stack symbol is '**b**' then state changes to 'q_2' and **no change** in top of stack.

Working with Stack :

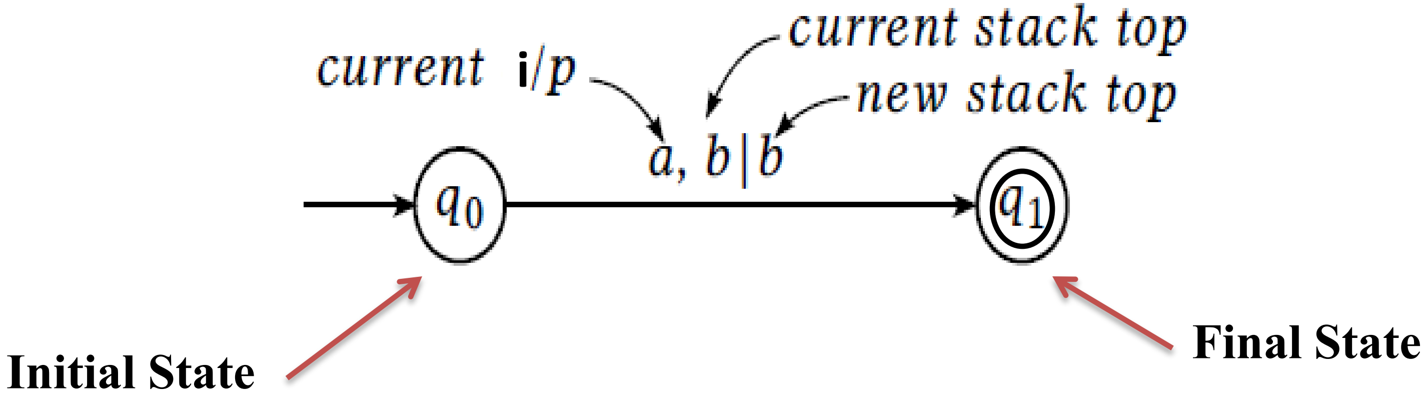


Transition :



Representation

Transition Diagram :



Representation

Transition Rule :

Current Stack Top **New Stack Top**

$$\delta(q0, a, b) = \{(q1, b)\}$$

Current State

New State

Current Input

Example 1 :

Q. Design PDA { $a^n b^n$, $n \geq 1$ }.

Language : { ab, aabb, aaabbb, ...}

Logic : For each input 'a', push 'a' into stack.

For each input 'b', pop one 'a' from stack

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

$\Gamma = \{ a, z_0 \}$

States :

q_s : initial state

q_0: read 'a' (push)

q_1: read 'b' (pop)

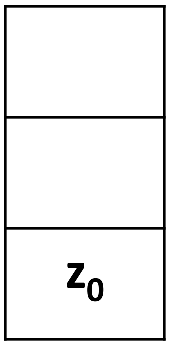
q_2: input is over and stack is empty (accept)

Initial state : q_s

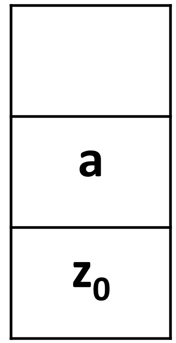
Finals state : q_2

Example Processing

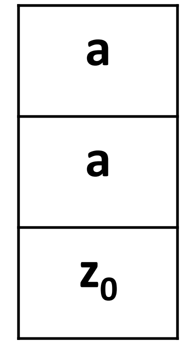
Input: a a b b



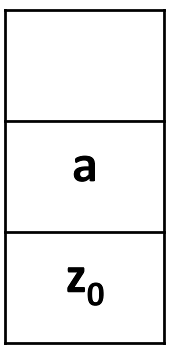
Initial Stack



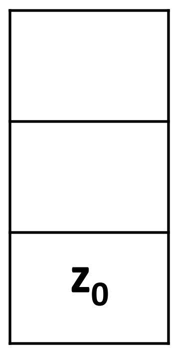
Push ‘a’



Push ‘a’



Pop ‘a’



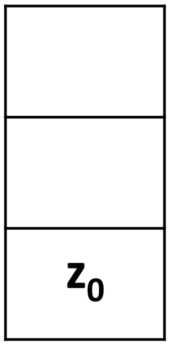
Pop ‘a’

Stack is empty and input is over so accept string

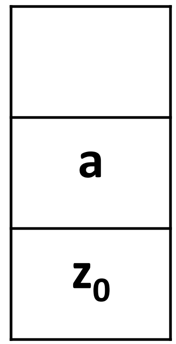
q_s : *initial state*
q_0: *read 'a' (push)*
q_1:*read 'b' (pop)*
q_2: *input is over and stack is empty (accept)*

Transition Rules

Input: a a b b

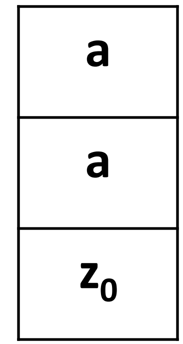


Initial Stack



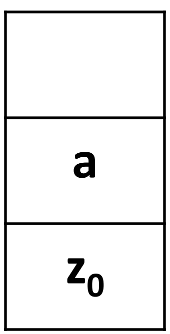
Push 'a'

$(q_s, a, z_0) = \{ (q_0, a\, z_0) \}$



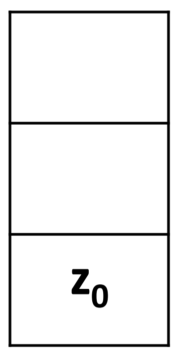
Push 'a'

$(q_o, a, a) = \{ (q_o, aa) \}$



Pop 'a'

$(q_o, b, a) = \{ (q_1, \varepsilon) \}$



Pop 'a'

$(q_1, b, a) = \{ (q_1, \varepsilon) \}$

Stack is empty and input is over

$(q_1, \varepsilon, z_0) = \{ (q_2, z_0) \}$

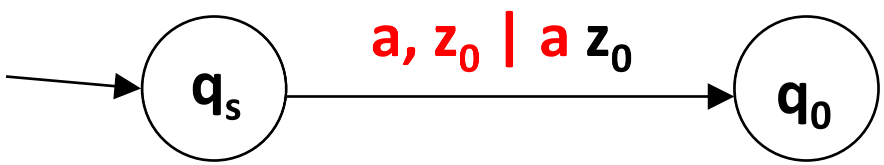
so accept string

Final Transition Rules

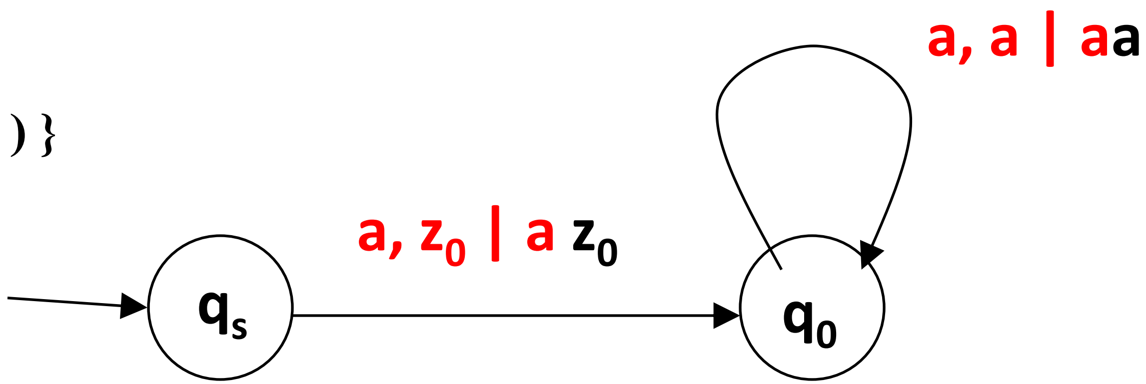
- $(q_s, a, z_0) = \{(q_0, az_0)\}$
- $(q_0, a, a) = \{(q_0, aa)\}$
- $(q_0, b, a) = \{(q_1, \varepsilon)\}$
- $(q_1, b, a) = \{(q_1, \varepsilon)\}$
- $(q_1, \varepsilon, z_0) = \{(q_2, z_0)\}$

Transition Diagram

- $(q_s, a, z_0) = \{ (q_0, \; a\, z_0) \}$

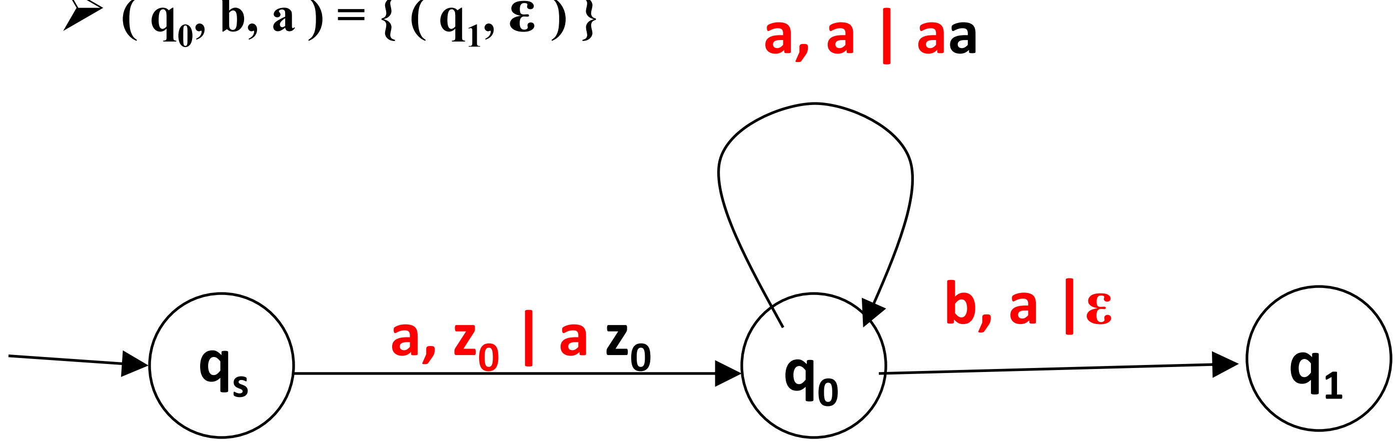


- $(q_0, a, a) = \{ (q_0, aa) \}$

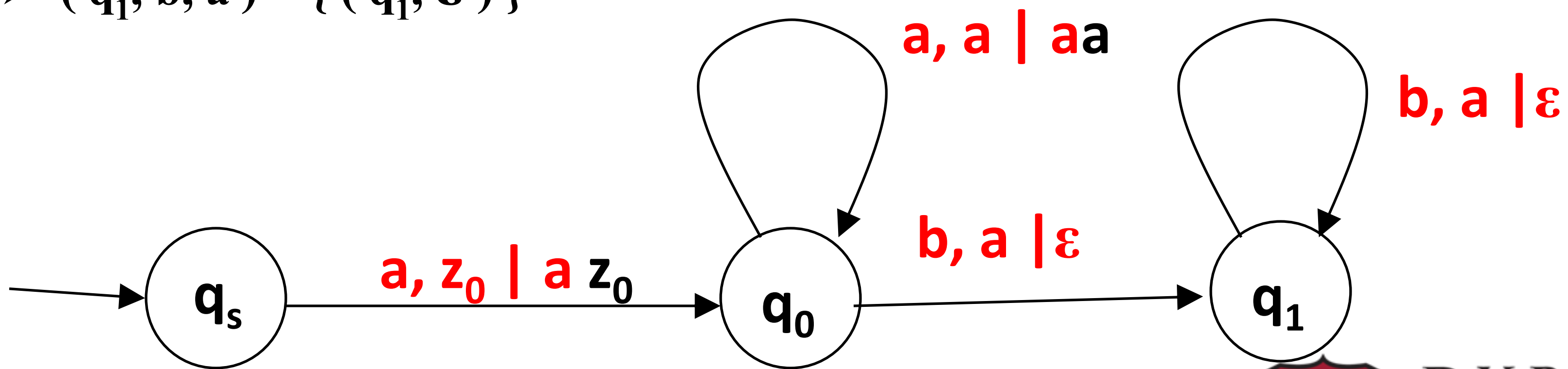


Transition Diagram (Cont..)

- $\delta(q_0, b, a) = \{ (q_1, \varepsilon) \}$

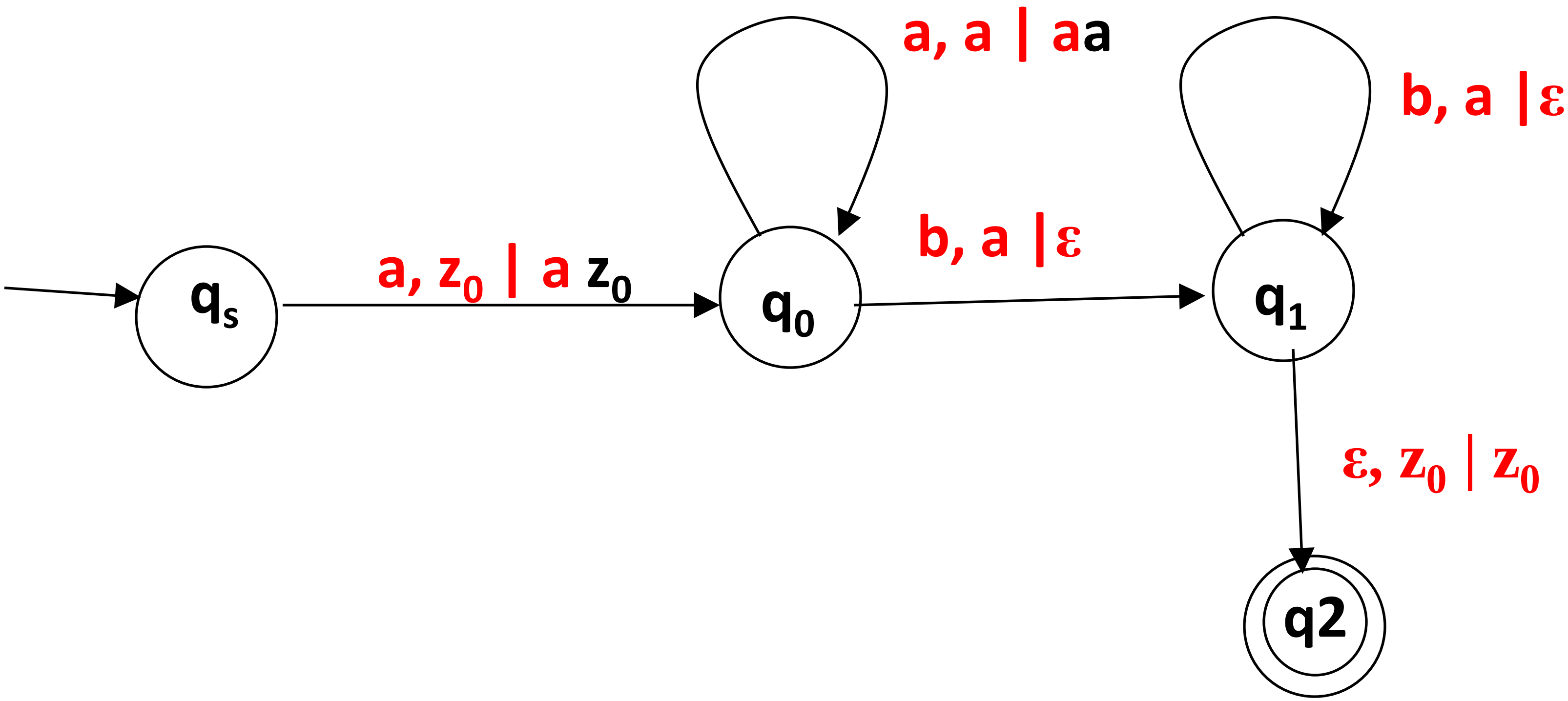


- $\delta(q_1, b, a) = \{ (q_1, \varepsilon) \}$



Transition Diagram (Cont..)

- $(q_1, \varepsilon, z_0) = \{ (q_2, z_0) \}$



D Y PATIL DEEMED TO BE UNIVERSITY RAMRAO ADIK INSTITUTE OF TECHNOLOGY NAVI MUMBAI

Simulation

Input : aabb

- **(q_s, aabb, z_0)**
- **(q_0, abb, a z_0)**
- **(q_0, bb, aa z_0)**
- **(q_1, b, a z_0)**
- **(q_1, ε, z_0)**
- **(q_2, z_0)   accept**

Example 2:

Q. Design PDA { $a^n b^{n+1}$, $n \geq 1$ }.

L = {abb, aabbb,aaabbbb,…………………}

Logic : For each input 'a', push ‘a' into stack.

For first ‘b’ perform no-operation

For remaining 'b', pop one ‘a' from stack

If input is over and stack is empty then accept

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

$\Gamma = \{ a, z_0 \}$

States:

q_s : *initial state*

q_0: *read ‘a’ (push)*

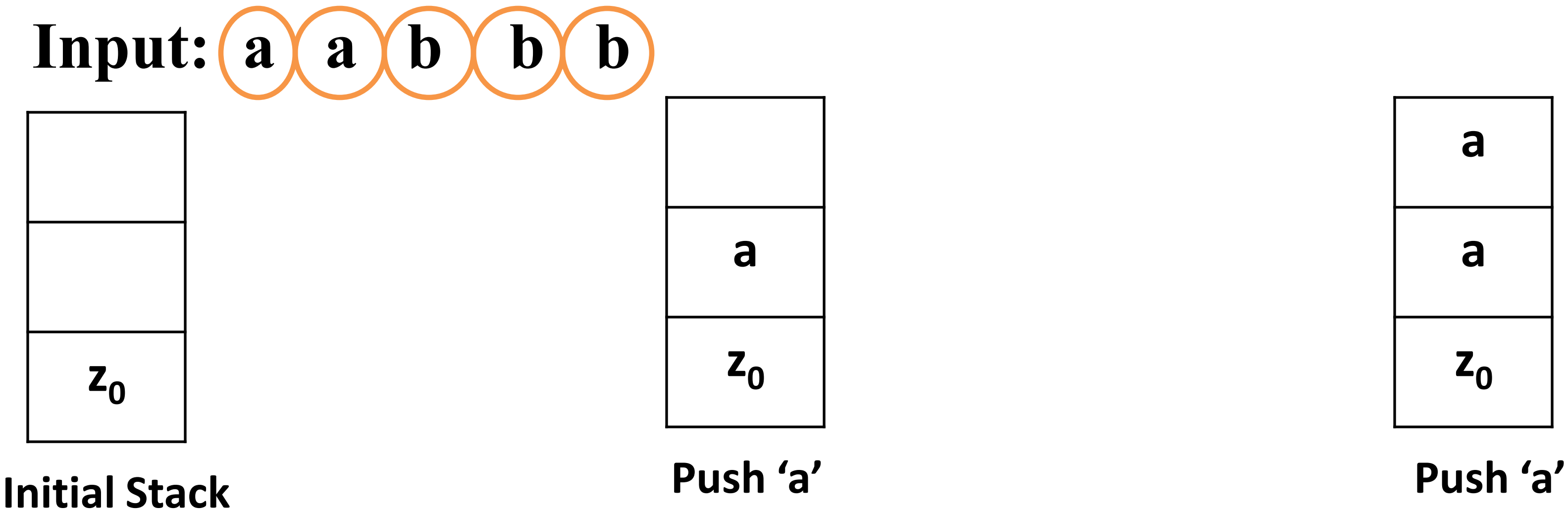
q_1: *read ‘b’ (pop) (except first ‘b’)*

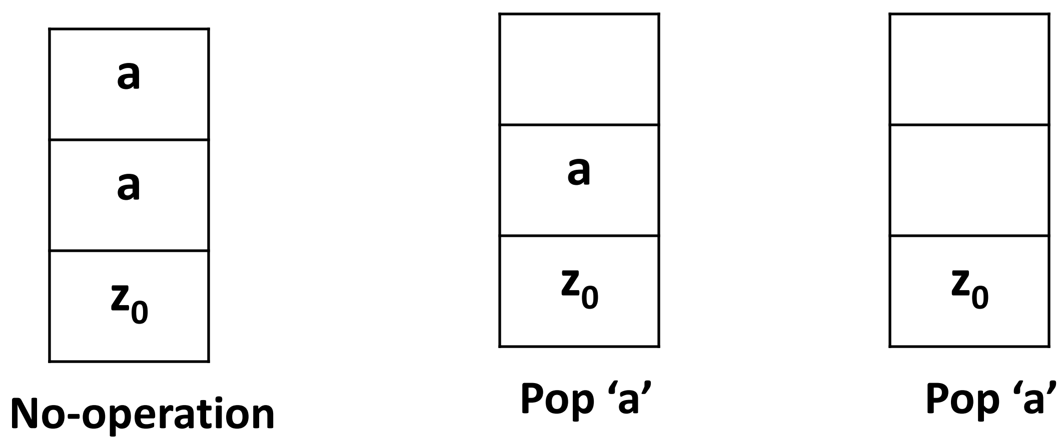
q_2: *input is over and stack is empty (accept)*

Initial state : q_s

Finals state: q_2

Example Processing





Stack is empty and input is over so accept string

Transition Rules

q_s : initial state
q_0: read 'a' (push)
q_1:read 'b' (pop) (except first 'b')
q_2: input is over and stack is empty (accept)

Input: a a b b b

		z_0

Initial Stack

	a	z_0

Push 'a'

$(q_s, a, z_0) = \{ (q_0, az_0) \}$

a	a	z_0

Push 'a'

$(q_0, a, a) = \{ (q_0, aa) \}$

a	a	z_0

No-op

$(q_0, b, a)=\{(q_1,a)\}$

a	a	z_0

Pop

$(q_1, b, a) = \{ (q_1, \varepsilon) \}$

	a	z_0

Pop

$(q_1, b, a) = \{ (q_1, \varepsilon) \}$

		z_0

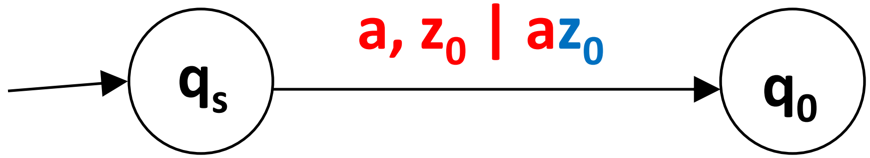
$(q_1, \varepsilon, z_0) = \{ (q_2, z_0) \}$

Final Transition Rules

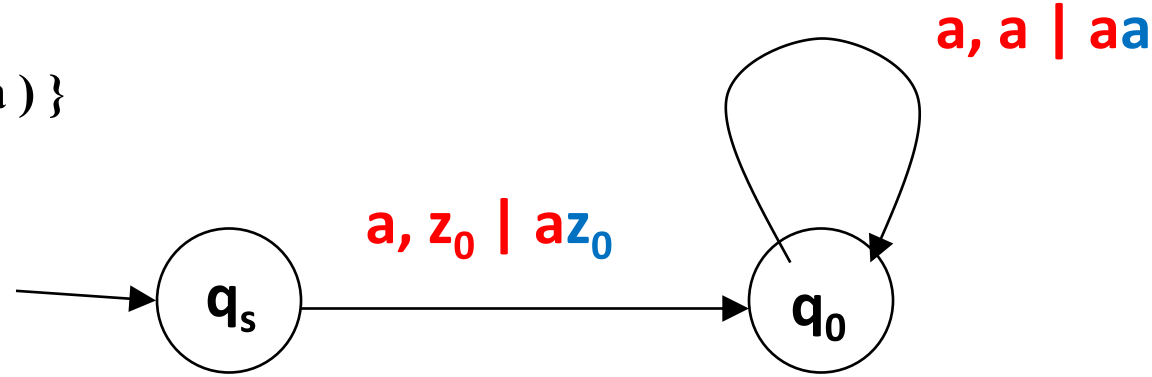
- $(q_s, a, z_0) = \{ (q_0, az_0) \}$
- $(q_0, a, a) = \{ (q_0, aa) \}$
- $(q_0, b, a) = \{ (q_1, a) \}$
- $(q_1, b, a) = \{ (q_1, \varepsilon) \}$
- $(q_1, \varepsilon, z_0) = \{ (q_2, z_0) \}$

Transition Diagram

- δ(q_s, a, z_0) = { (q_0, az_0) }

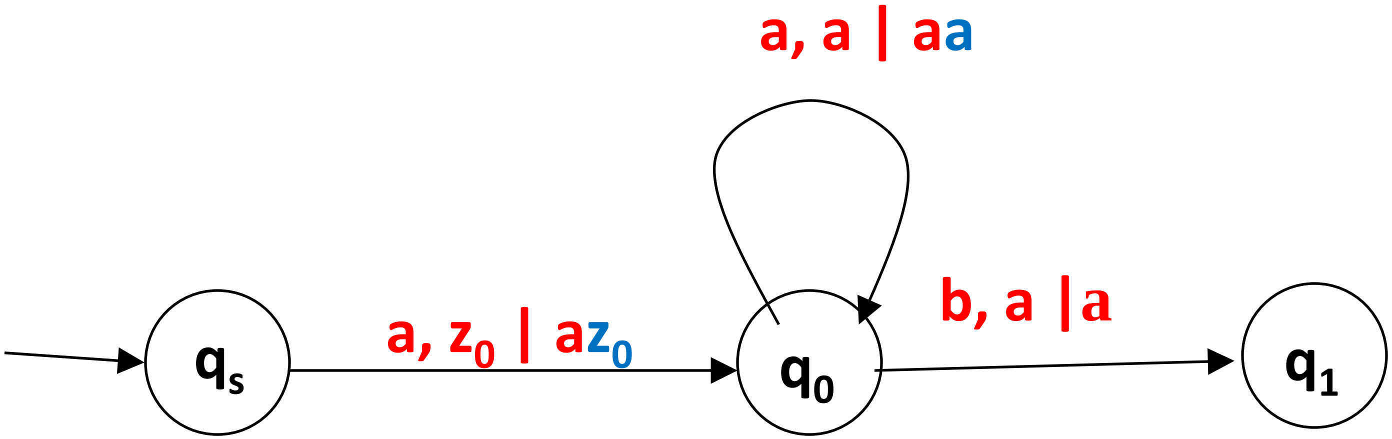


- δ(q_0, a, a) = { (q_0, aa) }

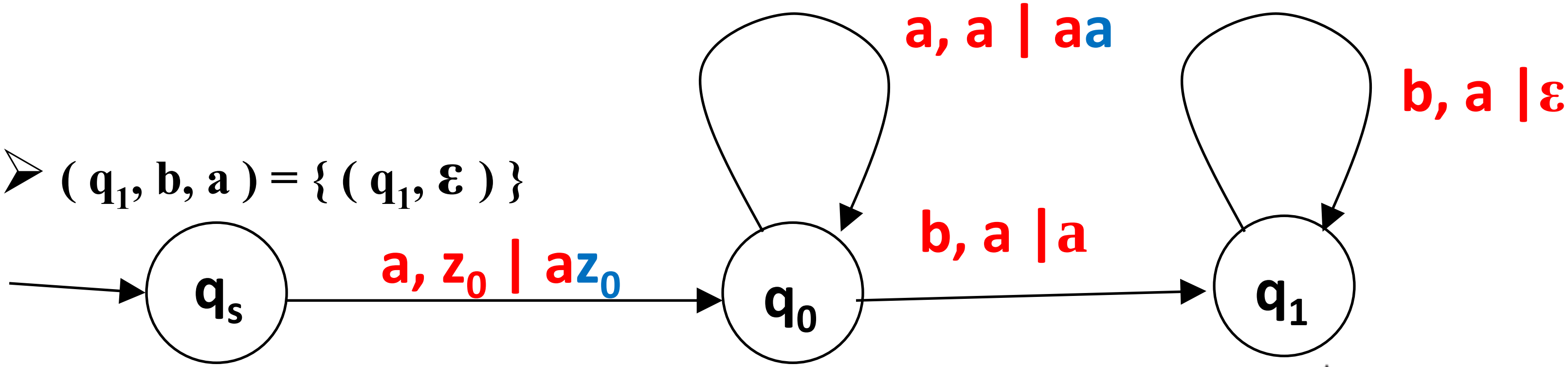


Transition Diagram (Cont..)

- $(q_0, b, a) = \{ (q_1, a) \}$

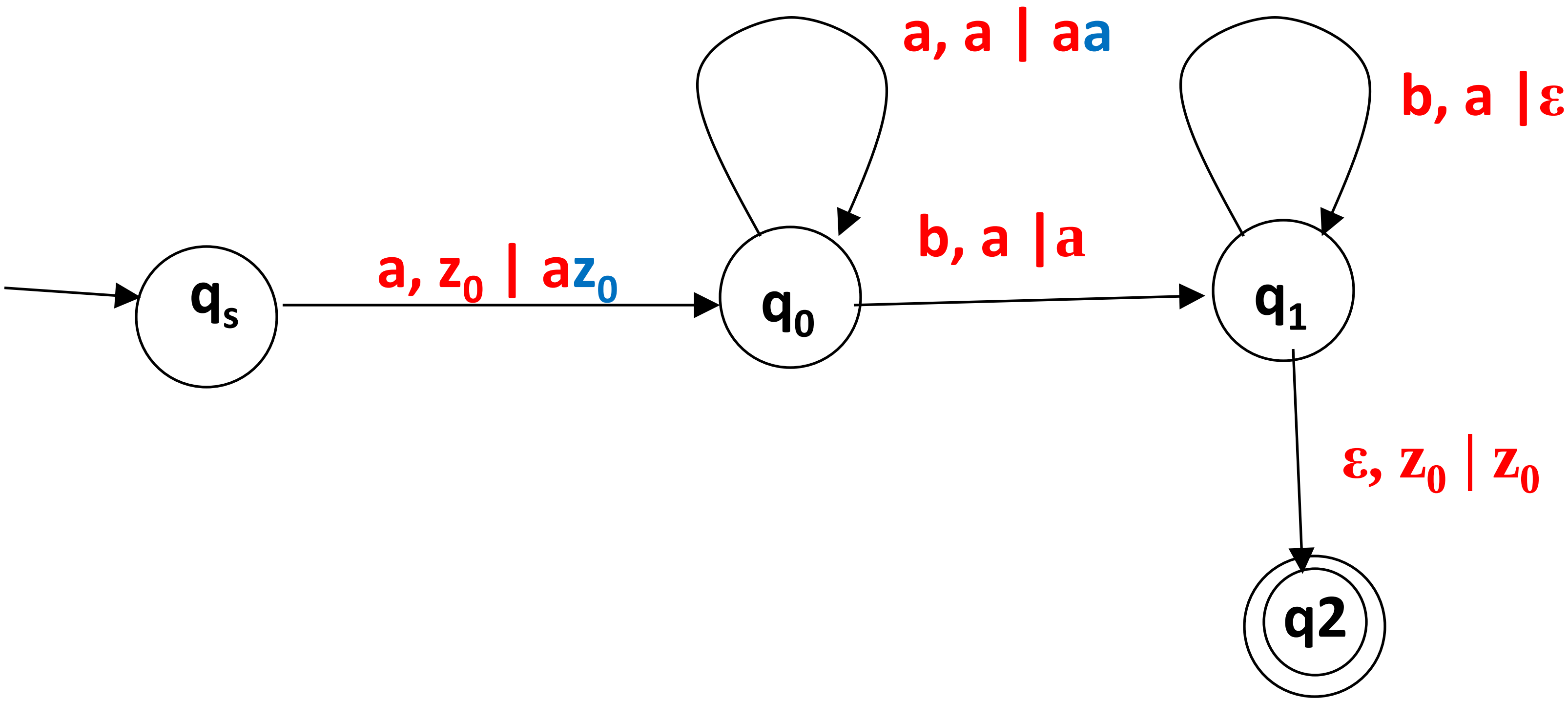


- $(q_1, b, a) = \{ (q_1, \varepsilon) \}$



Transition Diagram (Cont..)

- $(q_1, \varepsilon, z_0) = \{ (q_2, z_0) \}$



Simulation

Input: aabbb

- **$(q_s, \textcolor{red}{a}abbb, z_0)$**
- **$(q_0, \textcolor{red}{a}bbb, a\ z_0)$**
- **$(q_0, \textcolor{red}{b}bb, aa\ z_0)$**
- **$(q_1, \textcolor{red}{b}b, aa\ z_0)$**
- **$(q_1, \textcolor{red}{b}, a\ z_0)$**
- **(q_1, ε, z_0)**
- **(q_2, z_0)** **accept**

D Y PATIL
DEEMED TO BE
UNIVERSITY
RAMRAO ADIK
INSTITUTE OF TECHNOLOGY
NAVI MUMBAI

Thank You