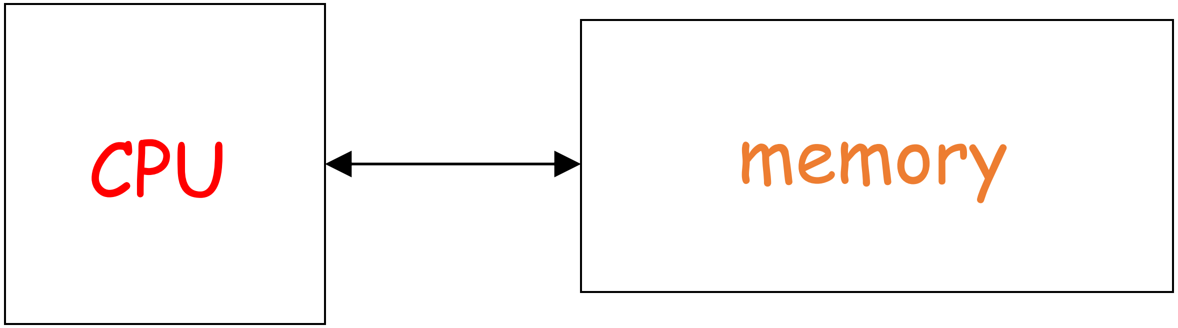
Module-1

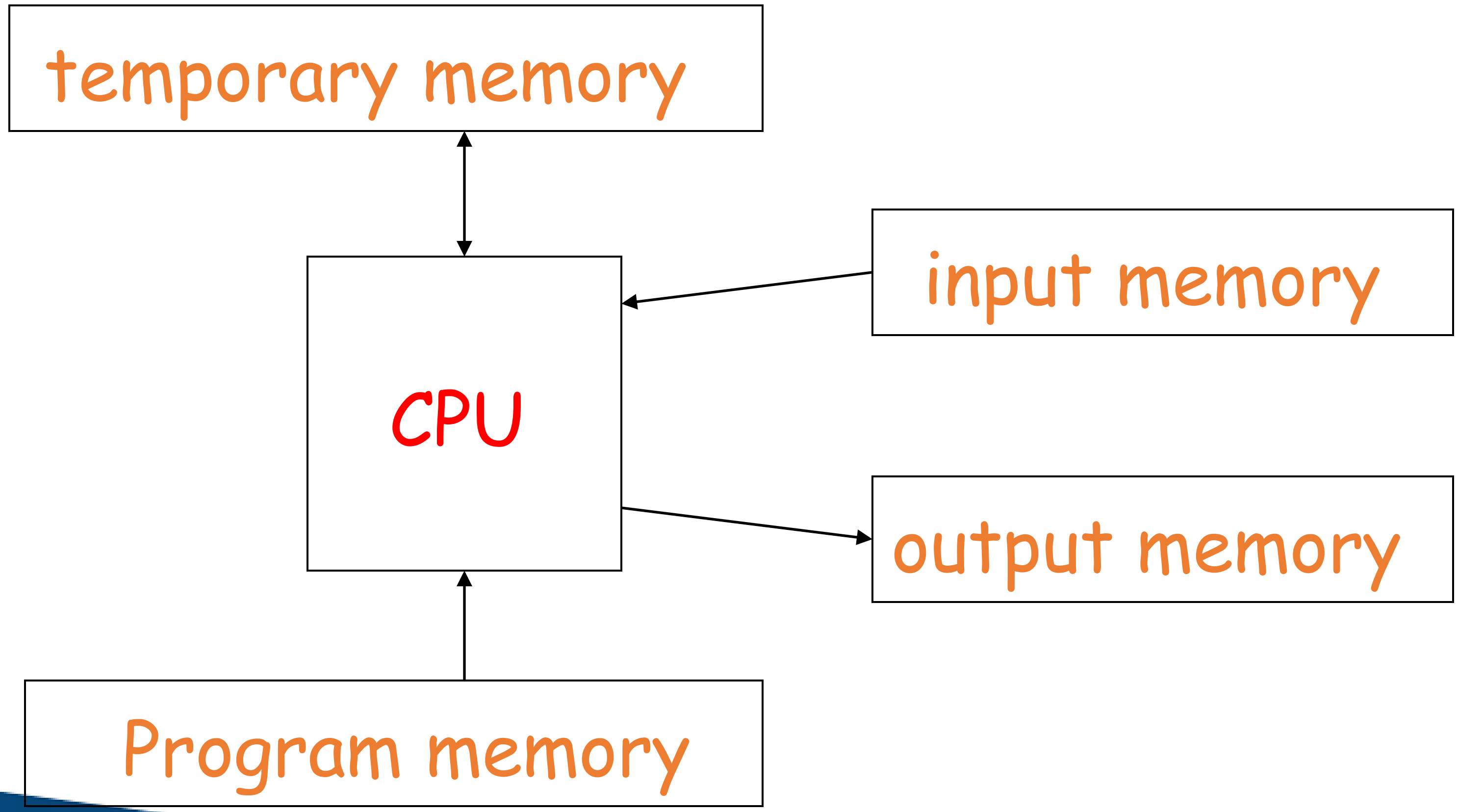
Introduction to Finite Automata

Private University Estd. in Karnataka State by Act No. 41 of 2013

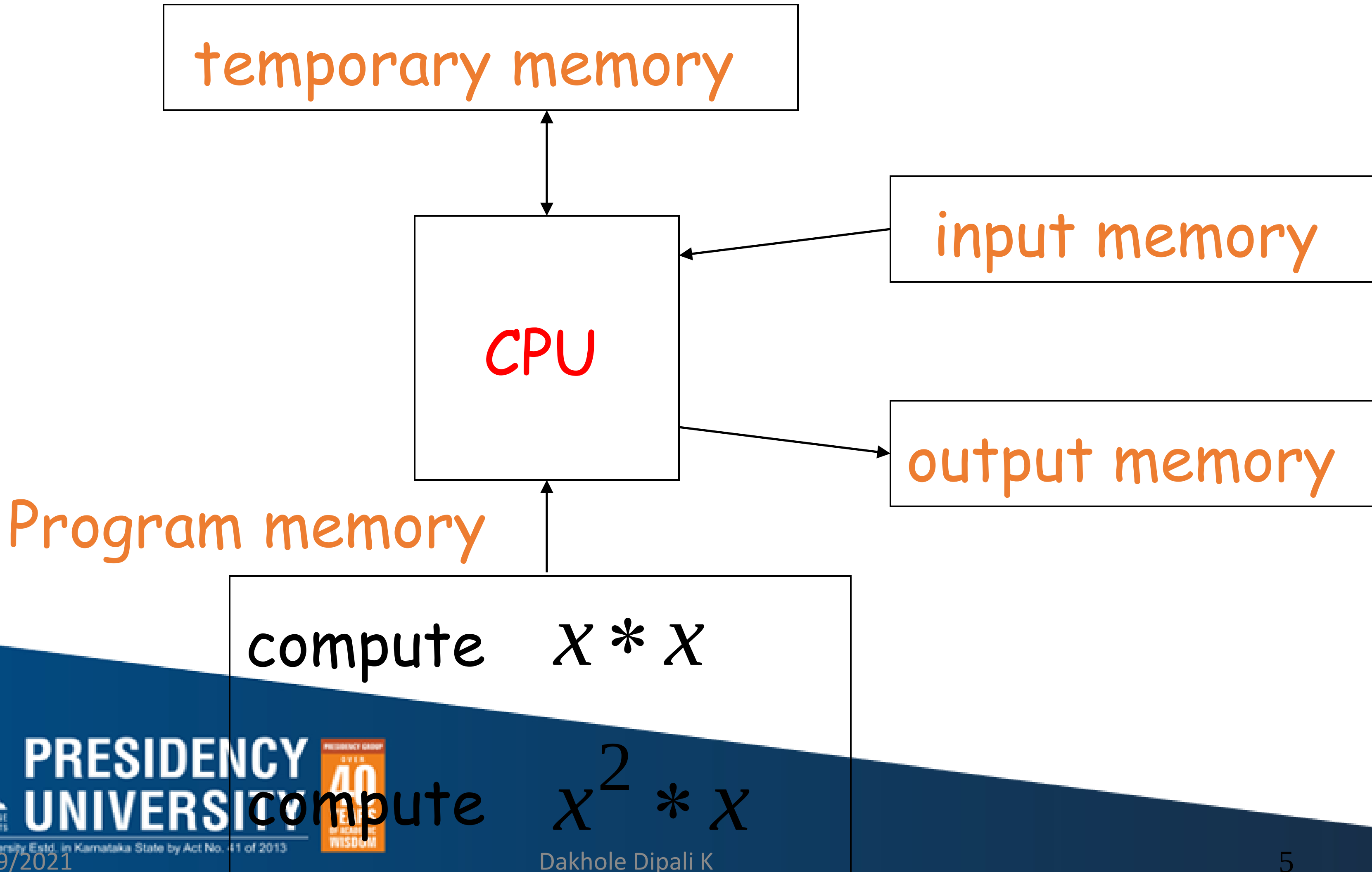
Models of Computation

Computation

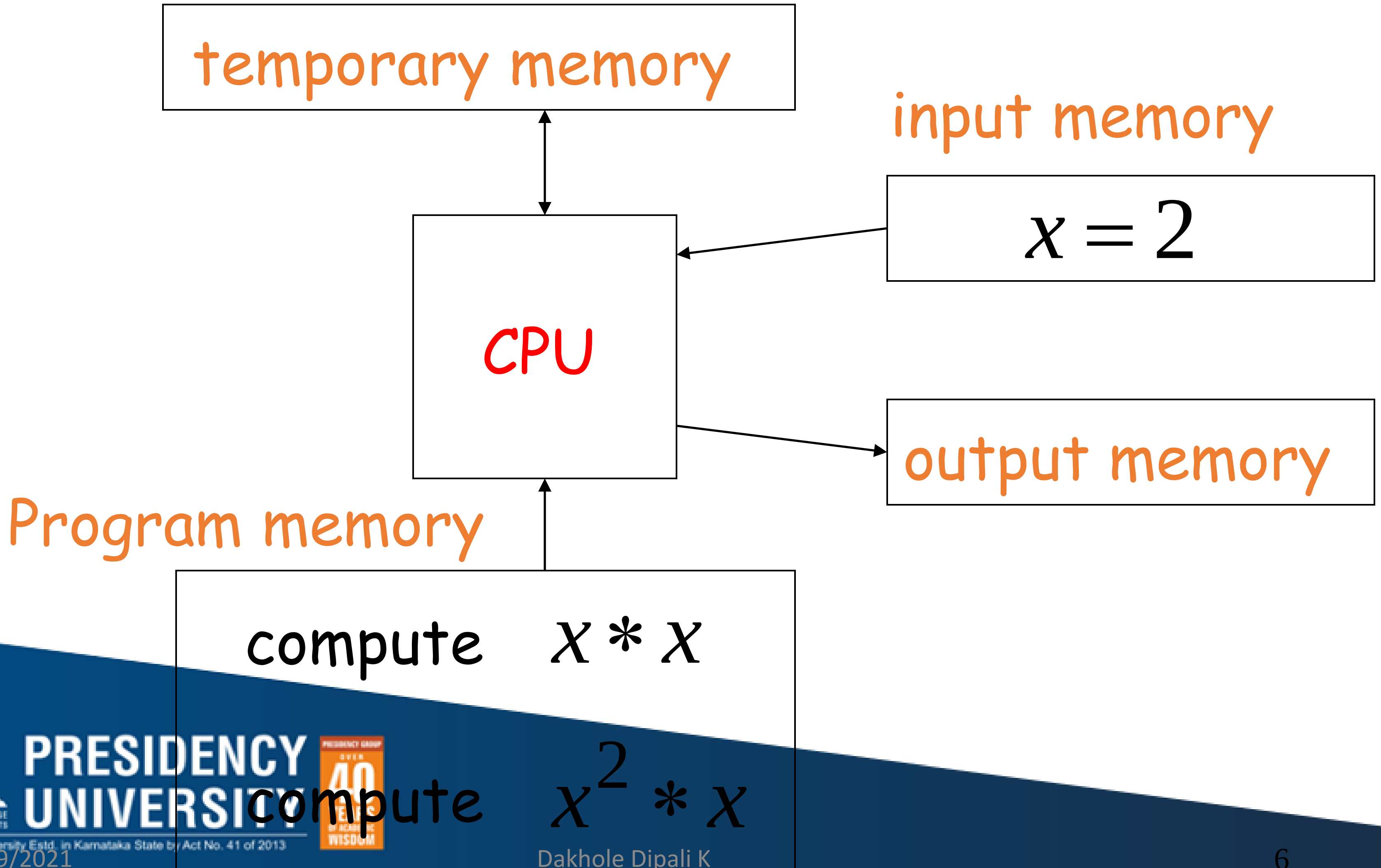



temporary memory
input memory
CPU
output memory
Program memory

Example: $f(x) = x^3$

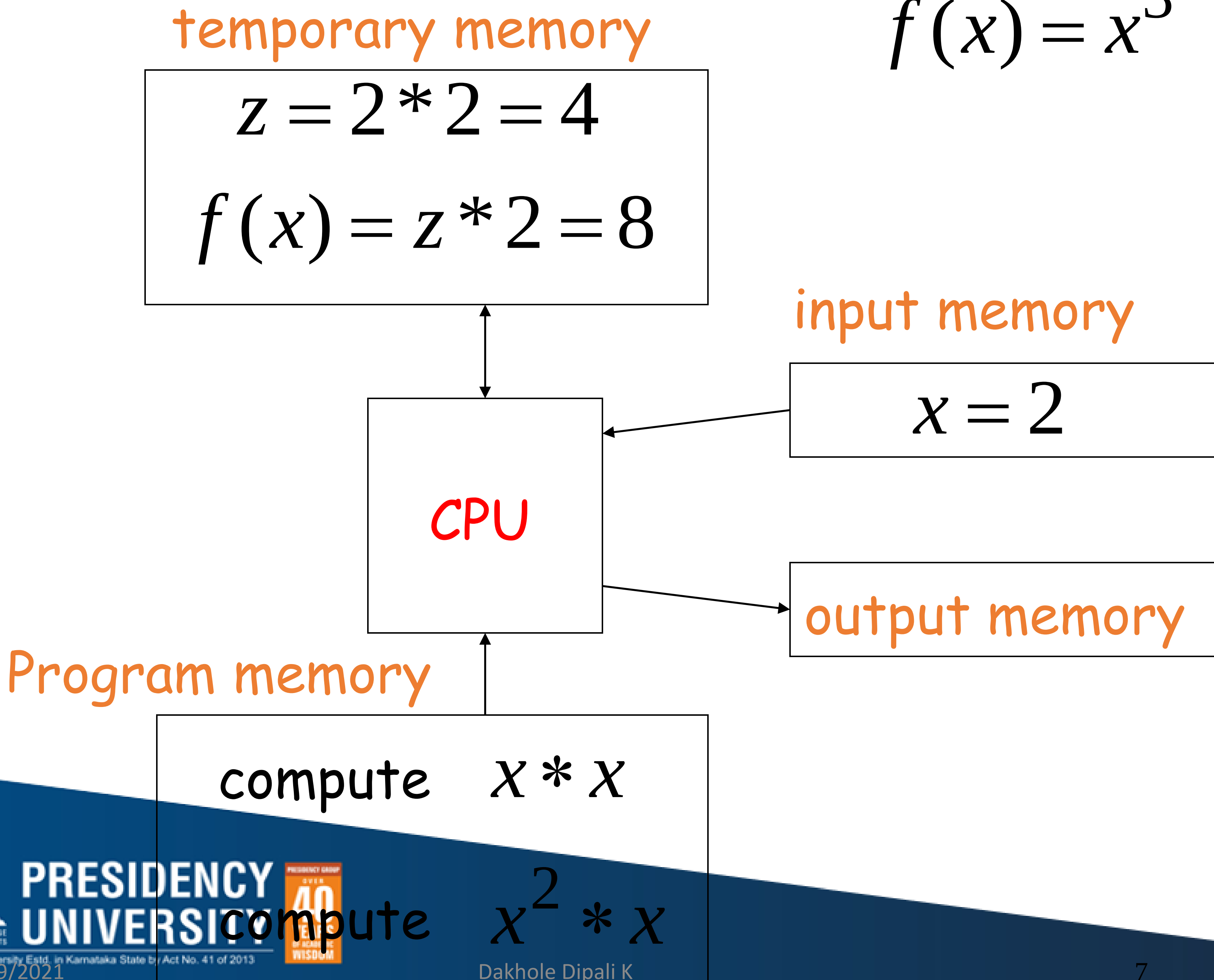


$$f(x) = x^3$$



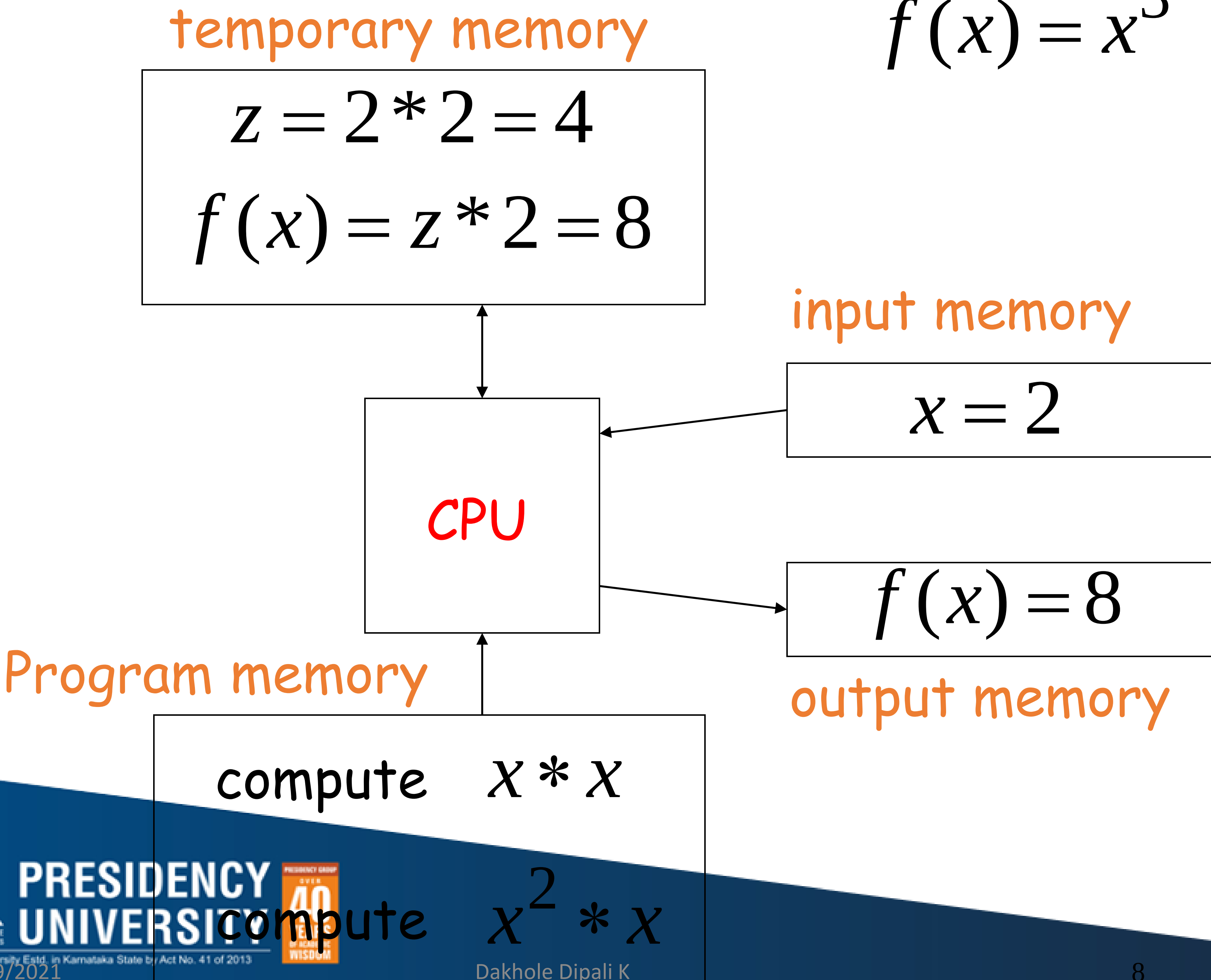
temporary memory
input memory
$x = 2$
CPU
output memory
Program memory
compute $x * x$
compute $x^2 * x$

$$f(x) = x^3$$



temporary memory
z = 2 * 2 = 4
f(x) = z * 2 = 8
input memory
x = 2
CPU
output memory
Program memory
compute x * x
compute x^2 * x

$$f(x) = x^3$$

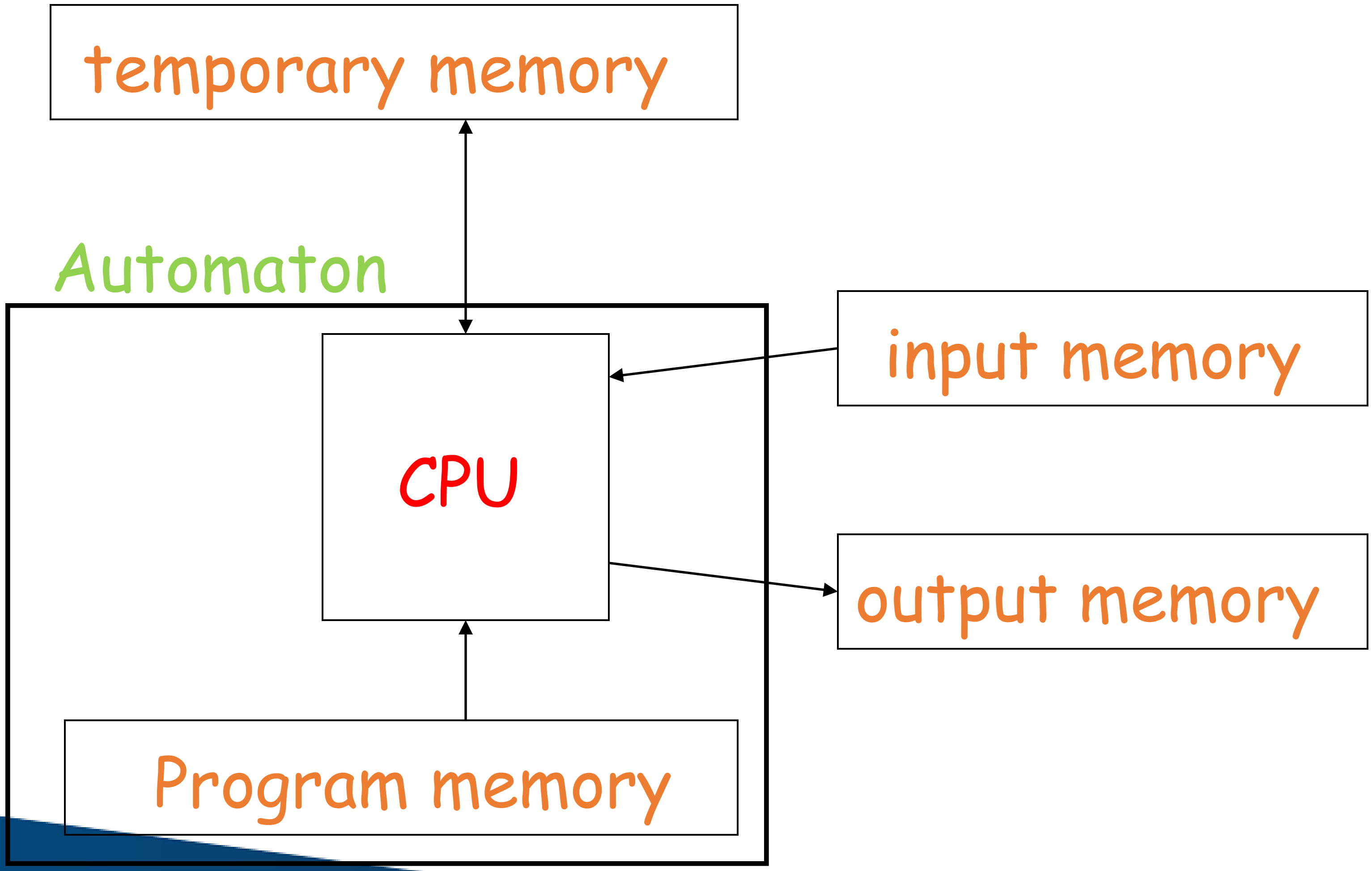

temporary memory
$z = 2*2 = 4$
$f(x) = z*2 = 8$
input memory
$x = 2$
CPU
$f(x) = 8$
output memory
Program memory
compute $x*x$
compute $x^2 * x$

Automata theory is the study of abstract machines and automata.

The computational problems can be solved using them.

It is a theory in theoretical computer science and discrete mathematics

Automaton



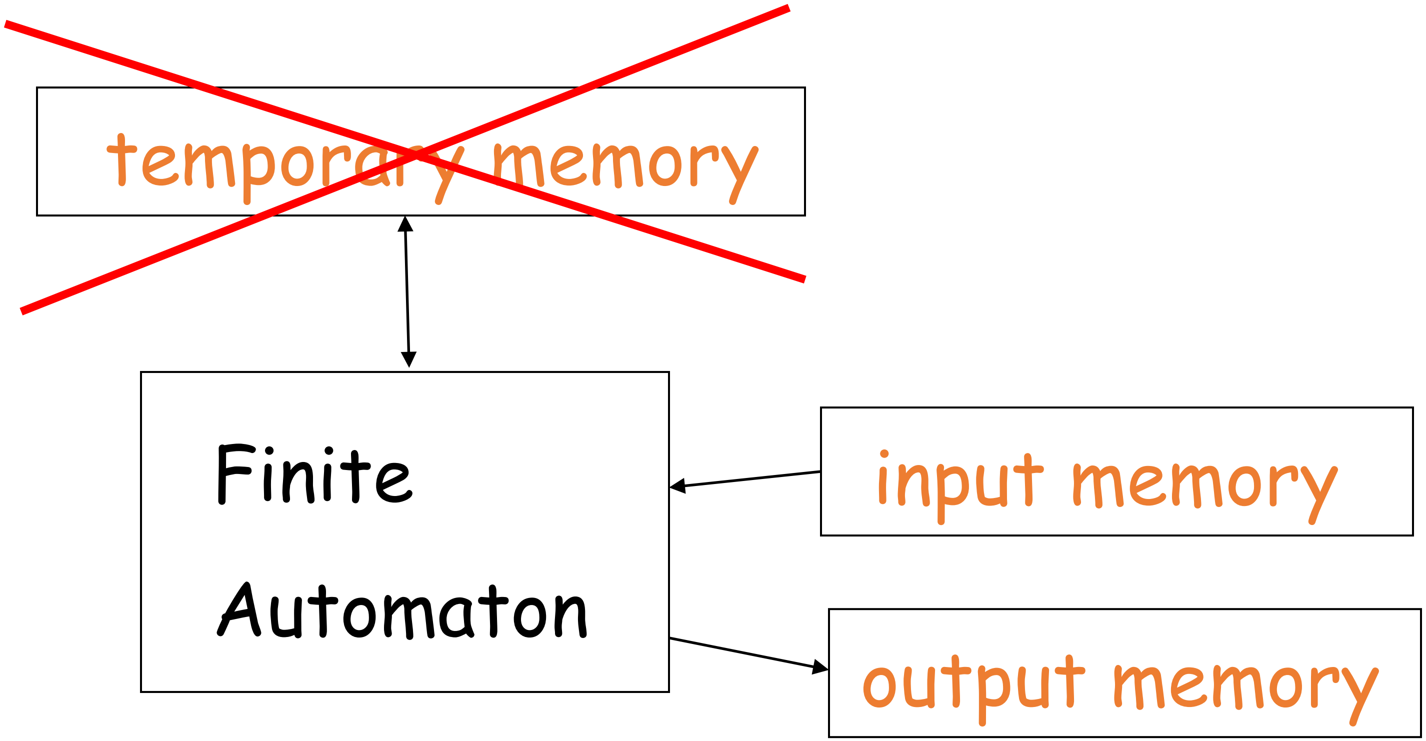
temporary memory
Automaton
CPU
input memory
output memory
Program memory

Different Kinds of Automata

Automata are distinguished by the temporary memory

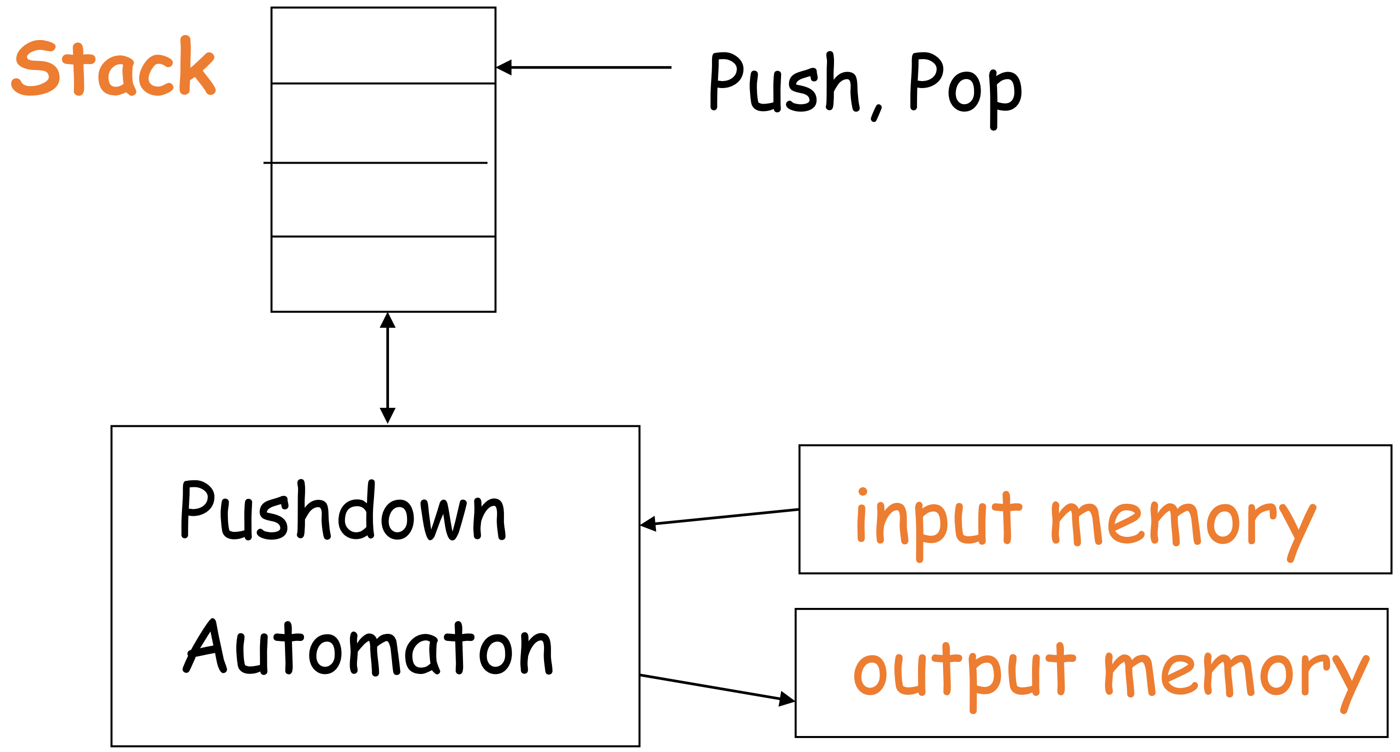
- **Finite Automata:** no temporary memory
- **Pushdown Automata:** stack
- **Turing Machines:** random access memory

Finite Automaton



Example: Vending Machines (small computing power)

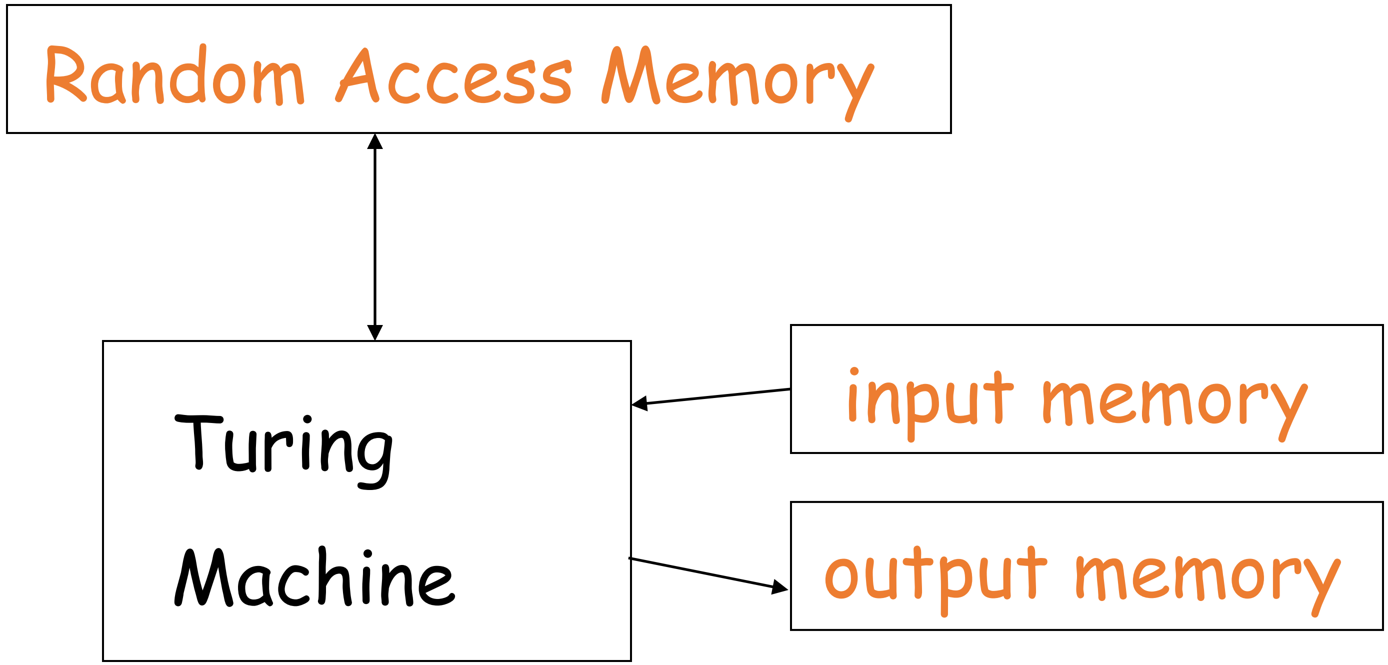
Pushdown Automaton



Example: Compilers for Programming Languages

(medium computing power)

Turing Machine



Examples: Any Algorithm (highest computing power)

Power of Automata

Finite Automata $<$ Pushdown Automata $<$ Turing Machine

Less power $\longrightarrow$ More power

Solve more computational problems

Applications of finite automata

- For the designing of lexical analysis of a compiler which breaks the input text into logical units like identifiers, keywords etc.
- For recognizing the pattern using regular expressions.
- For the designing of the combination and sequential circuits.
- Software for designing and checking the behavior of digital circuits.
- Used in text editors.
- For the implementation of spell checkers.
- Software for scanning large bodies of text like web pages to find occurrence of words, phrases and other patterns.
- Software to verify all types that have finite number of distinct states such as communications protocols for secure exchange of information.

Applications of Push Down Automata

- For designing the parsing phase of a compiler (Syntax Analysis).
- For implementation of stack applications.
- For evaluating the arithmetic expressions.
- For solving the Tower of Hanoi Problem.

Applications of Turing machine

- For solving any recursively enumerable problem.
- For understanding complexity theory.
- For implementation of neural networks.
- For implementation of Robotics Applications.
- For implementation of artificial intelligence.

General Concepts of Automata Theory

- Symbol
- Alphabet
- Strings
- Empty Strings
- Length of the string
- Power of an Alphabet
- Concatenation of two strings
- Languages

Alphabets and Strings

- Symbol: a, b, 1 ,2, 3, 0, etc.

- We will use small alphabets: $\Sigma = \{a, b\}$

- Strings

$$a$$
$$ab$$
$$abba$$
$$baba$$
$$aaabbbaabab$$

$$u = ab$$
$$v = bbbaaa$$
$$w = abba$$

String Operations

$$w = a_1 a_2 \cdots a_n \qquad abba$$

$$v = b_1 b_2 \cdots b_m \qquad bbbaaa$$

1. Concatenation

$$wv = a_1 a_2 \cdots a_n b_1 b_2 \cdots b_m \qquad abbabbbaaa$$

2. Reverse

$$w = a_1 a_2 \cdots a_n$$

$$abab aaabbb$$

$$w^R = a_n \cdots a_2 a_1$$

$$bbbaaababa$$

3. String Length

$$w = a_1 a_2 \cdots a_n$$

Length: $|w| = n$

Examples:

$$|abba| = 4$$

$$|aa| = 2$$

$$|a| = 1$$

4. Length of Concatenation

$$|uv| = |u| + |v|$$

- Example:

$$u = aab, \quad |u| = 3$$
$$v = abaab, \quad |v| = 5$$

$$|uv| = |aababaab| = 8$$
$$|uv| = |u| + |v| = 3 + 5 = 8$$

5. Empty String

- A string with no letters: λ or ϵ

- Observations: $|\lambda| = 0$

$$\lambda w = w\lambda = w$$

$$\lambda abba = abba\lambda = abba$$

6. Substring

- Substring of string:
  - a subsequence of consecutive characters

String	Substring
$abbab$	ab
$\underline{ab}bab$	$abba$
$\underline{abba}b$	b
$ab\underline{b}ab$	$bbab$

7. Prefix and Suffix

$$abbab$$

Prefixes	Suffixes
λ	$abbab$
a	$bbab$
ab	bab
abb	ab
$abba$	b
$abbab$	λ

$$w = uv$$

prefix

suffix

PRESIDENCY UNIVERSITY

Private University Estd. in Karnataka State by Act No. 41 of 2013

8. Another Operation

$$w^n = \underbrace{ww \cdots w}_{n}$$

- Example: $(abba)^2 = abbaabba$

- Definition: $w^0 = \lambda$

$$(abba)^0 = \lambda$$

9. The * Operation

Σ^* the set of all possible strings from Σ alphabet

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

$$\Sigma^* = \{\lambda, a, b, aa, ab, ba, bb, aaa, aab, \ldots\}$$

10. The + Operation

Σ^+ : the set of all possible strings from alphabet Σ except λ

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

$$\Sigma^* = \{\lambda, a, b, aa, ab, ba, bb, aaa, aab, \ldots\}$$

$$\boxed{\Sigma^+ = \Sigma^* - \lambda}$$

$$\Sigma^+ = \{a, b, aa, ab, ba, bb, aaa, aab, \ldots\}$$

Languages

- A language is a set of strings

- String: A sequence of letters
  - Examples: "cat", "dog", "house", …
  - Defined over an alphabet:

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

- A language is any subset of Σ^*

- Example:

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

$$\Sigma^* = \{\lambda, a, b, aa, ab, ba, bb, aaa, \ldots\}$$

- Languages:

$$\{\lambda\}$$

$$\{a, aa, aab\}$$

$$\{\lambda, abba, baba, aa, ab, aaaaaa\}$$

Note that

Sets $\varnothing = \{\ \} \neq \{\lambda\}$

Set size $|\{\ \}| = |\varnothing| = 0$

Set size $|\{\lambda\}| = 1$

String length $|\lambda| = 0$

Another Example

An infinite language

$$L = \{a^n b^n : n \geq 0\}$$

$$\left.\begin{array}{l} \lambda \\ ab \\ aabb \\ aaaaabbbbb \end{array}\right\} \in L \qquad abb \notin L$$

Operations on Languages

1. The usual set operations

$$\{a, ab, aaaa\} \cup \{bb, ab\} = \{a, ab, bb, aaaa\}$$

$$\{a, ab, aaaa\} \cap \{bb, ab\} = \{ab\}$$

$$\{a, ab, aaaa\} - \{bb, ab\} = \{a, aaaa\}$$

Complement: $\overline{L} = \Sigma^* - L$

$$\overline{\{a, ba\}} = \{\lambda, b, aa, ab, bb, aaa, \ldots\}$$

2. Reverse

Definition:
$$L^R = \{w^R : w \in L\}$$

Examples:
$$\{ab, aab, baba\}^R = \{ba, baa, abab\}$$

$$L = \{a^n b^n : n \geq 0\}$$

$$L^R = \{b^n a^n : n \geq 0\}$$

PRESIDENCY UNIVERSITY
GAIN MORE KNOWLEDGE REACH GREATER HEIGHTS
Private University Estd. in Karnataka State by Act No. 41 of 2013
PRESIDENCY GROUP OVER 40 YEARS OF ACADEMIC WISDOM

3.Concatenation

- Definition: $L_1L_2 = \{xy : x \in L_1, y \in L_2\}$

- Example: $\{a, ab, ba\}\{b, aa\}$

$$= \{ab, aaa, abb, abaa, bab, baaa\}$$

4. Another Operation

- Definition: $$L^n = \underbrace{LL \cdots L}_{n}$$

$$\{a,b\}^3 = \{a,b\}\{a,b\}\{a,b\} =$$
$$\{aaa, aab, aba, abb, baa, bab, bba, bbb\}$$

- Special case: $$L^0 = \{\lambda\}$$

$$\{a, bba, aaa\}^0 = \{\lambda\}$$

More Examples

$$L = \{a^n b^n : n \geq 0\}$$

$$L^2 = \{a^n b^n a^m b^m : n, m \geq 0\}$$

$$aabbaaabbb \in L^2$$

5. Star-Closure (Kleene *)

Definition: $L^* = L^0 \cup L^1 \cup L^2 \cdots$

- Example:

$$\{a, bb\}^* = \left\{ \begin{array}{l} \lambda, \\ a, bb, \\ aa, abb, bba, bbbb, \\ aaa, aabb, abba, abbbb, \ldots \end{array} \right\}$$

PRESIDENCY UNIVERSITY
GAIN MORE KNOWLEDGE REACH GREATER HEIGHTS
Private University Estd. in Karnataka State by Act No. 41 of 2013
PRESIDENCY GROUP OVER 40 YEARS OF ACADEMIC WISDOM

6. Positive Closure

Definition: $$L^+ = L^1 \cup L^2 \cup \cdots$$
$$= L^* - \{\lambda\}$$

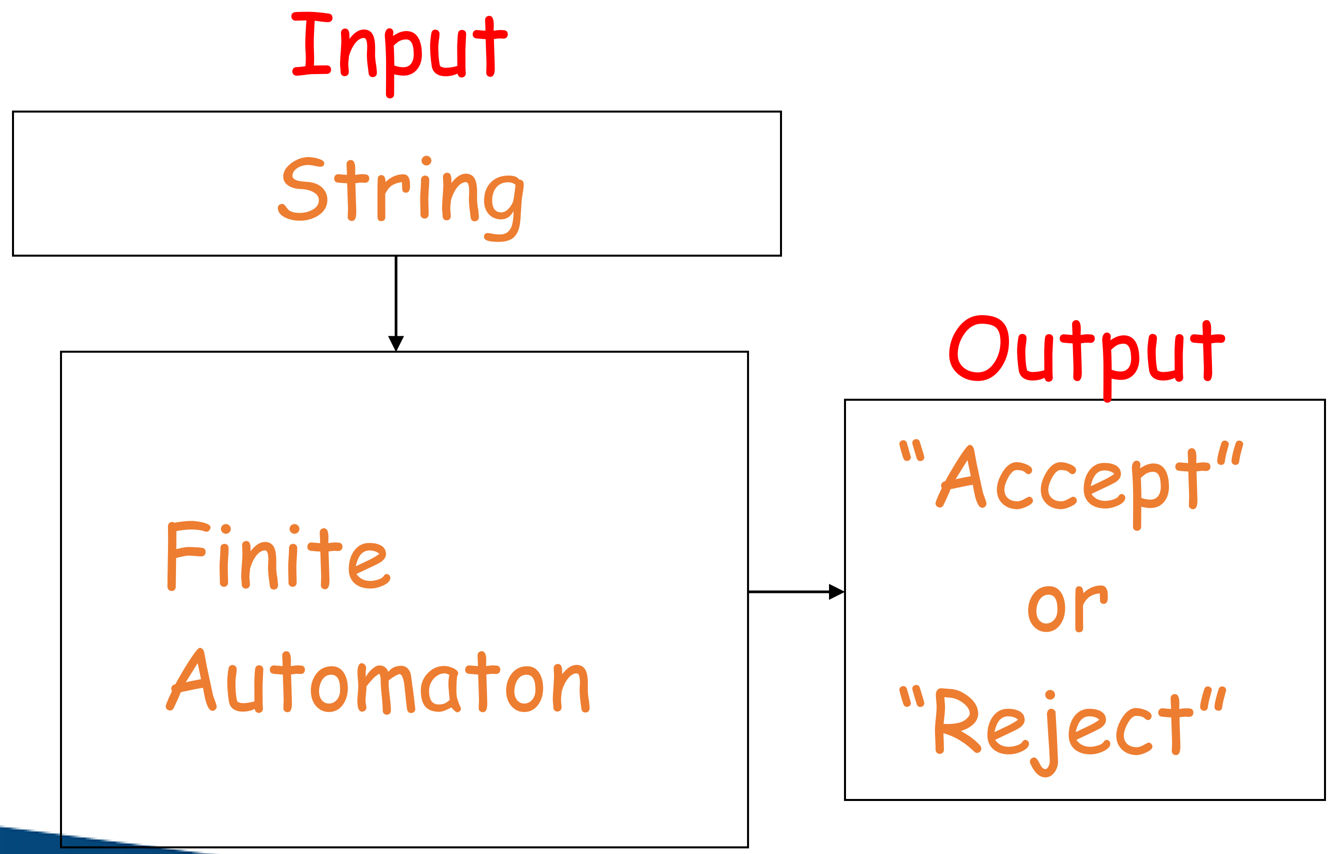
$$\{a, bb\}^+ = \left\{ \begin{array}{l} a, bb, \\ aa, abb, bba, bbbb, \\ aaa, aabb, abba, abbbb, \ldots \end{array} \right\}$$

Language Recognizers : An example of Finite Automata

An automaton is an abstract model of a digital computer.

Finite Automata(FA) is the simplest machine to recognize patterns.

Finite Automaton

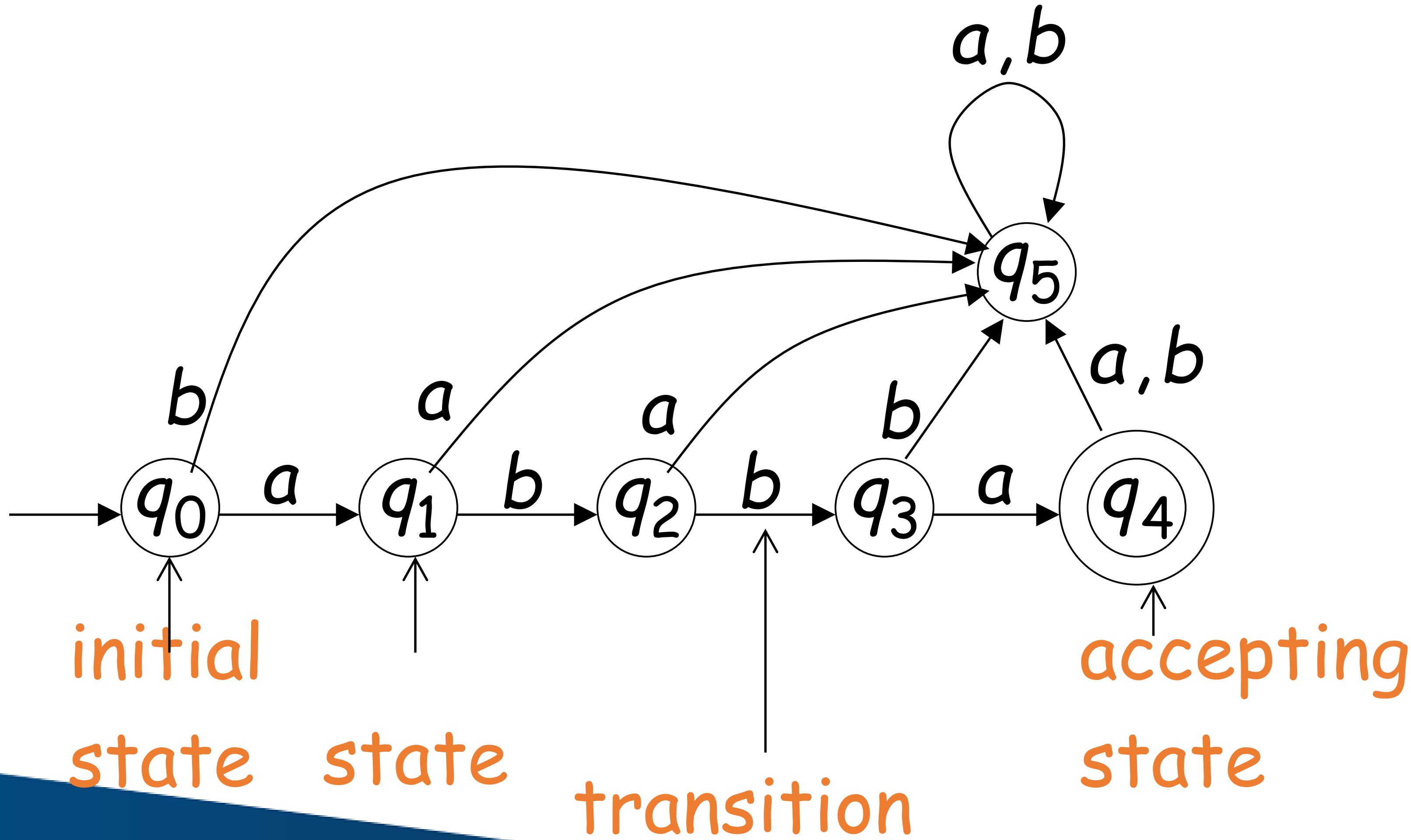


Representation of Finite Automata

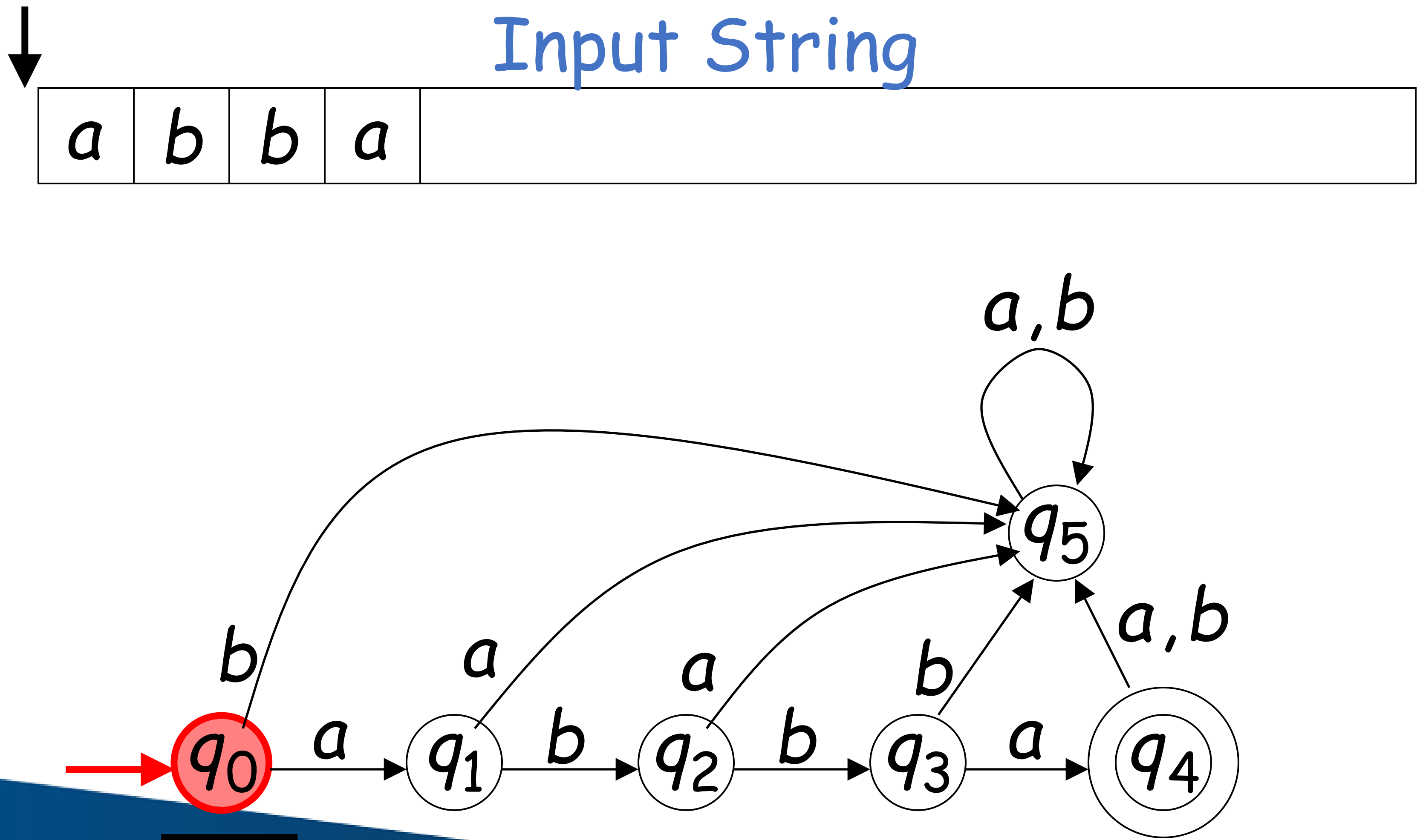
Finite Automata is represented by –

1. Transition Graph
2. Transition Table
3. Regular Expression

Transition Graph

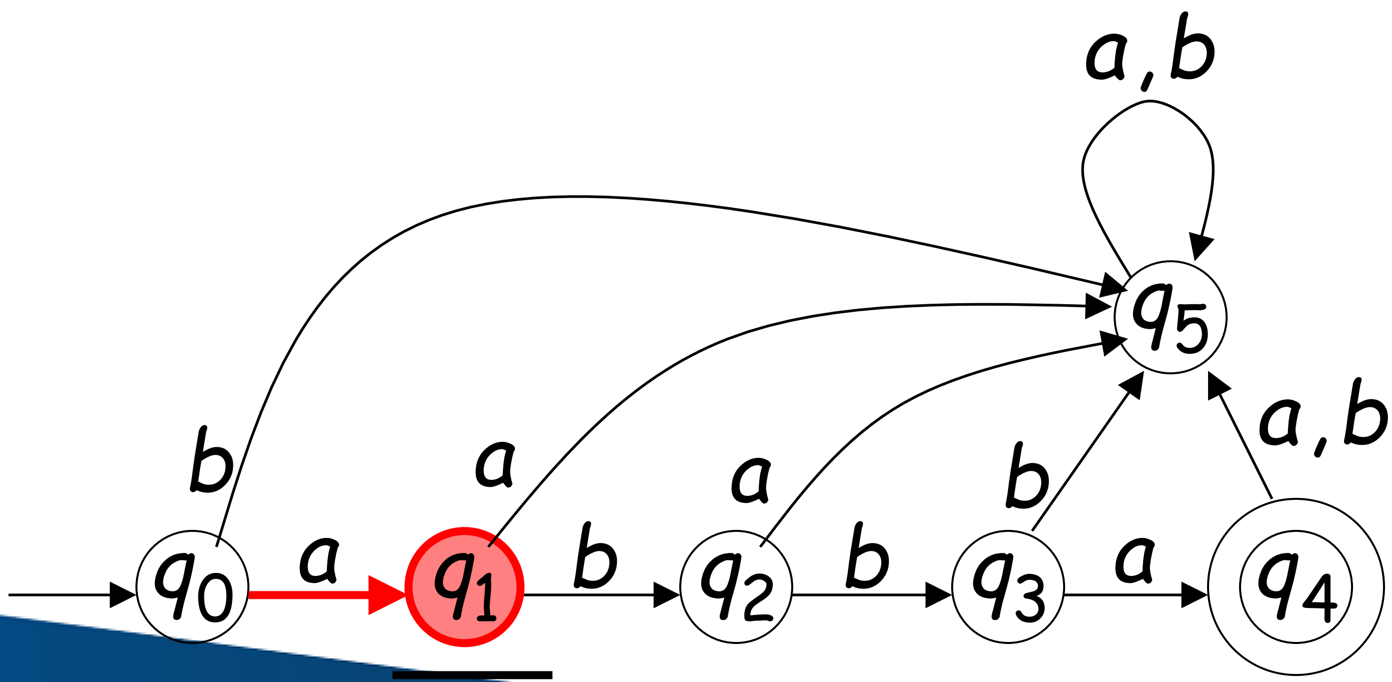


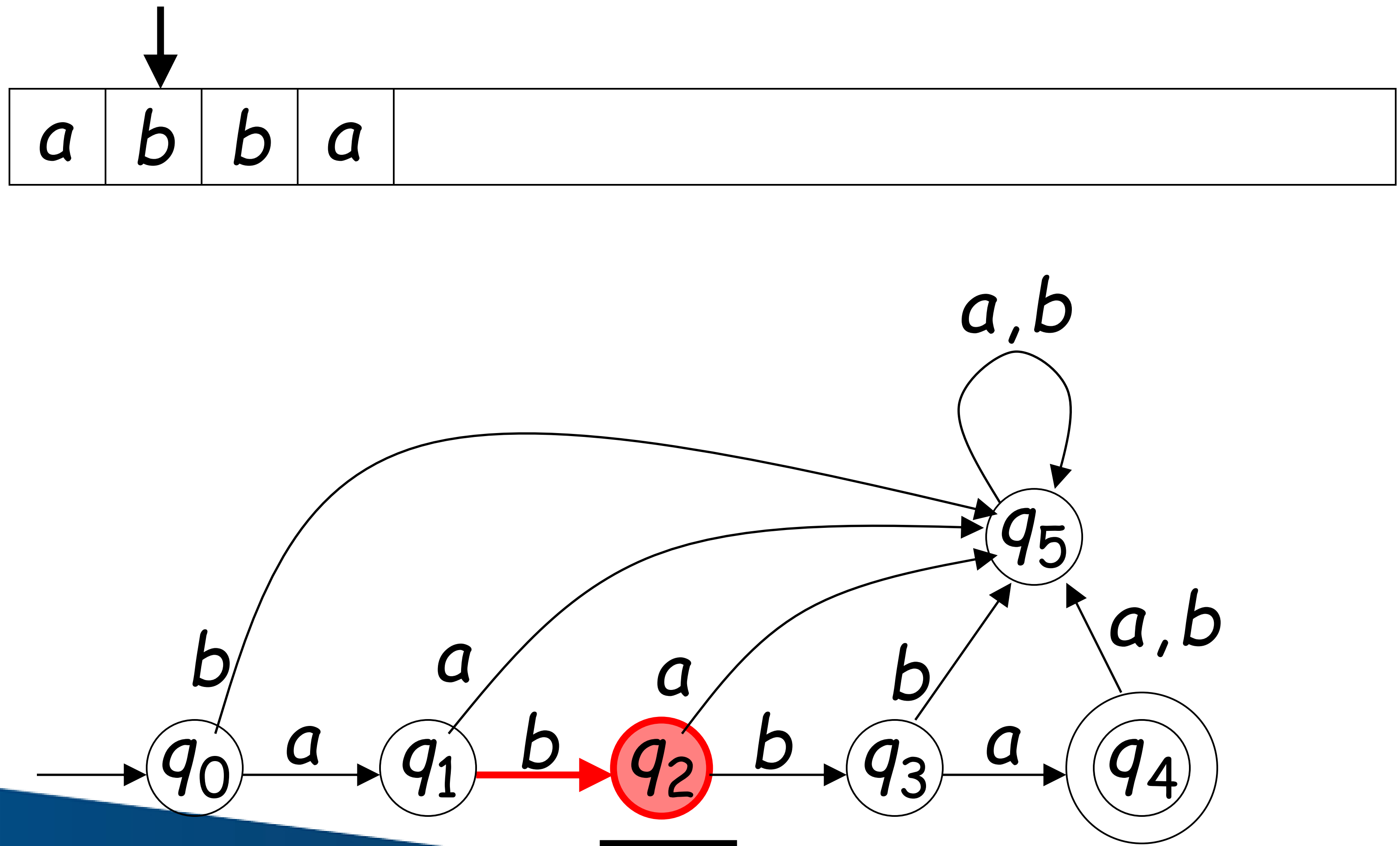
Initial Configuration

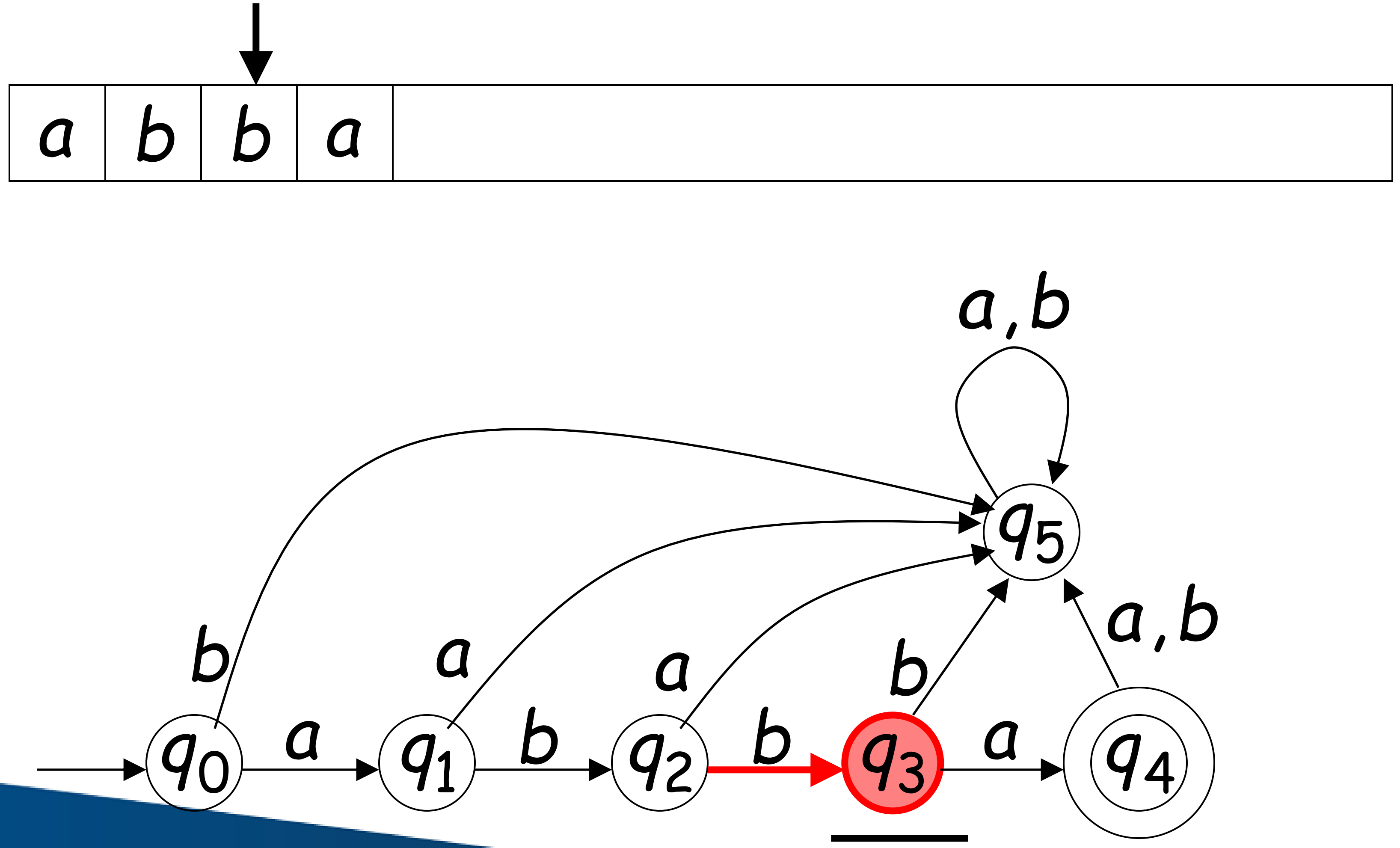


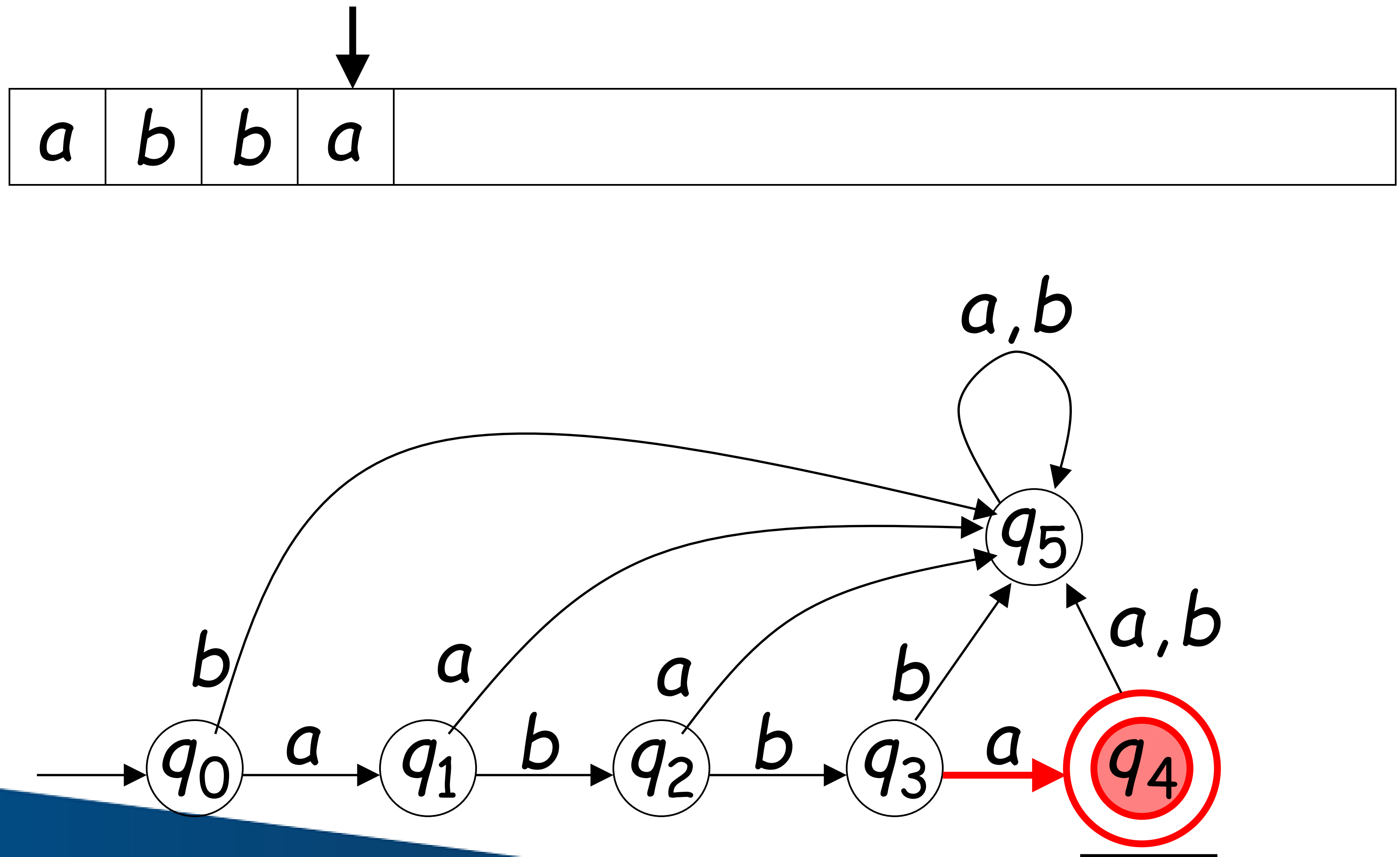
Reading the Input

a	b	b	a	

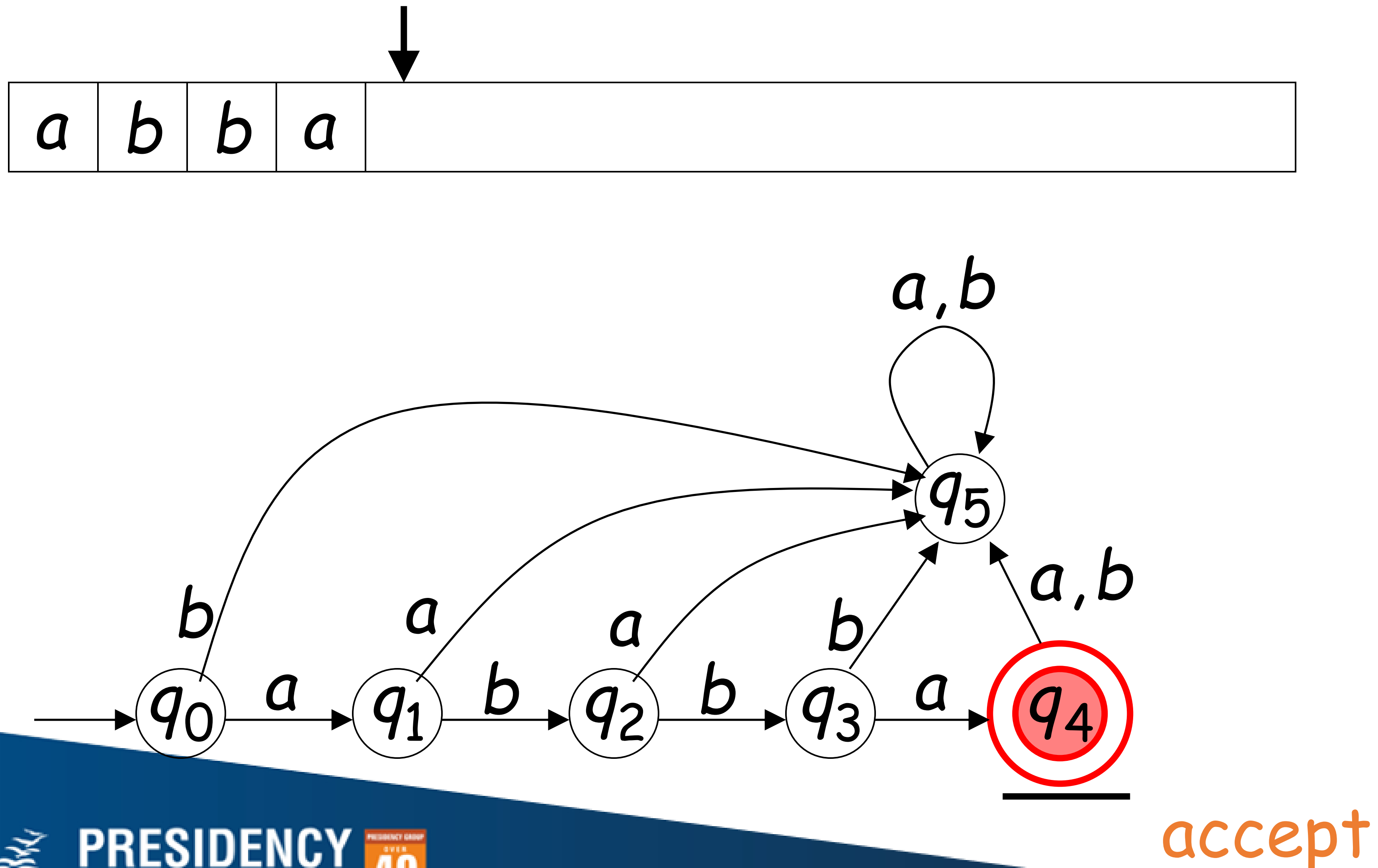



a
b
b
a
a,b
q_5
b
a
a
a,b
b
q_0
a
q_1
b
q_2
b
q_3
a
q_4


a
b
b
a
a,b
q5
b
a
a
b
a,b
q0
a
q1
b
q2
b
q3
a
q4

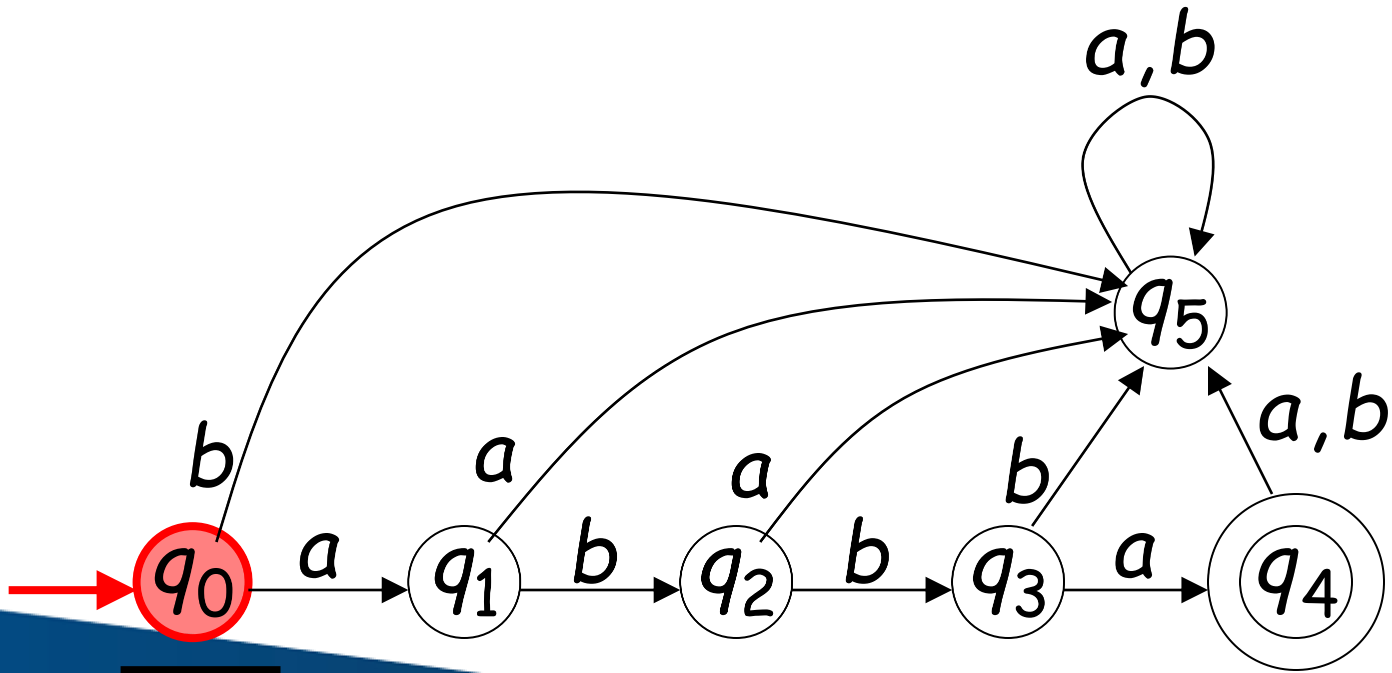

a
b
b
a
a,b
b
a
a
b
a,b
q_0
a
q_1
b
q_2
b
q_3
a
q_4
q_5

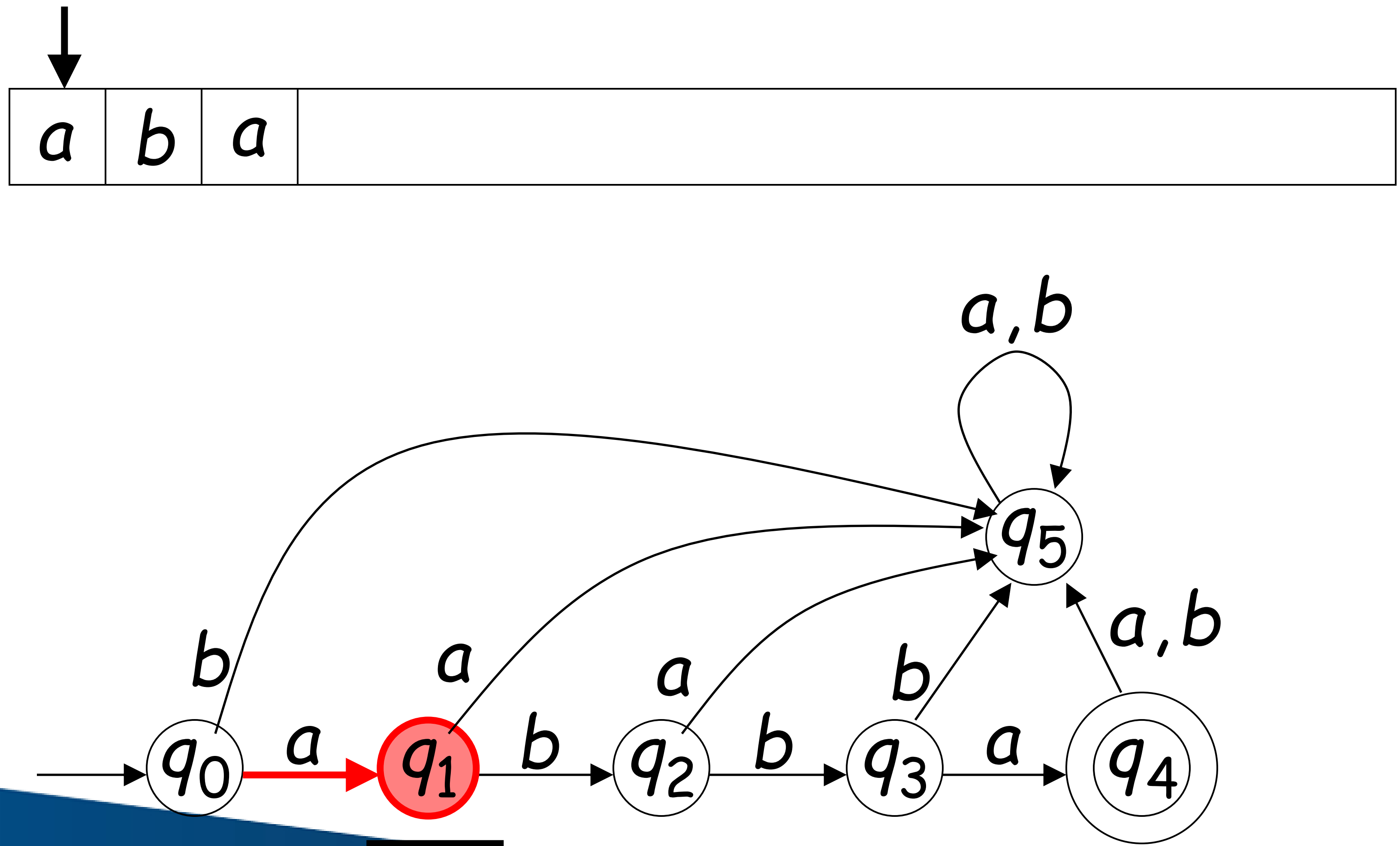
Input finished



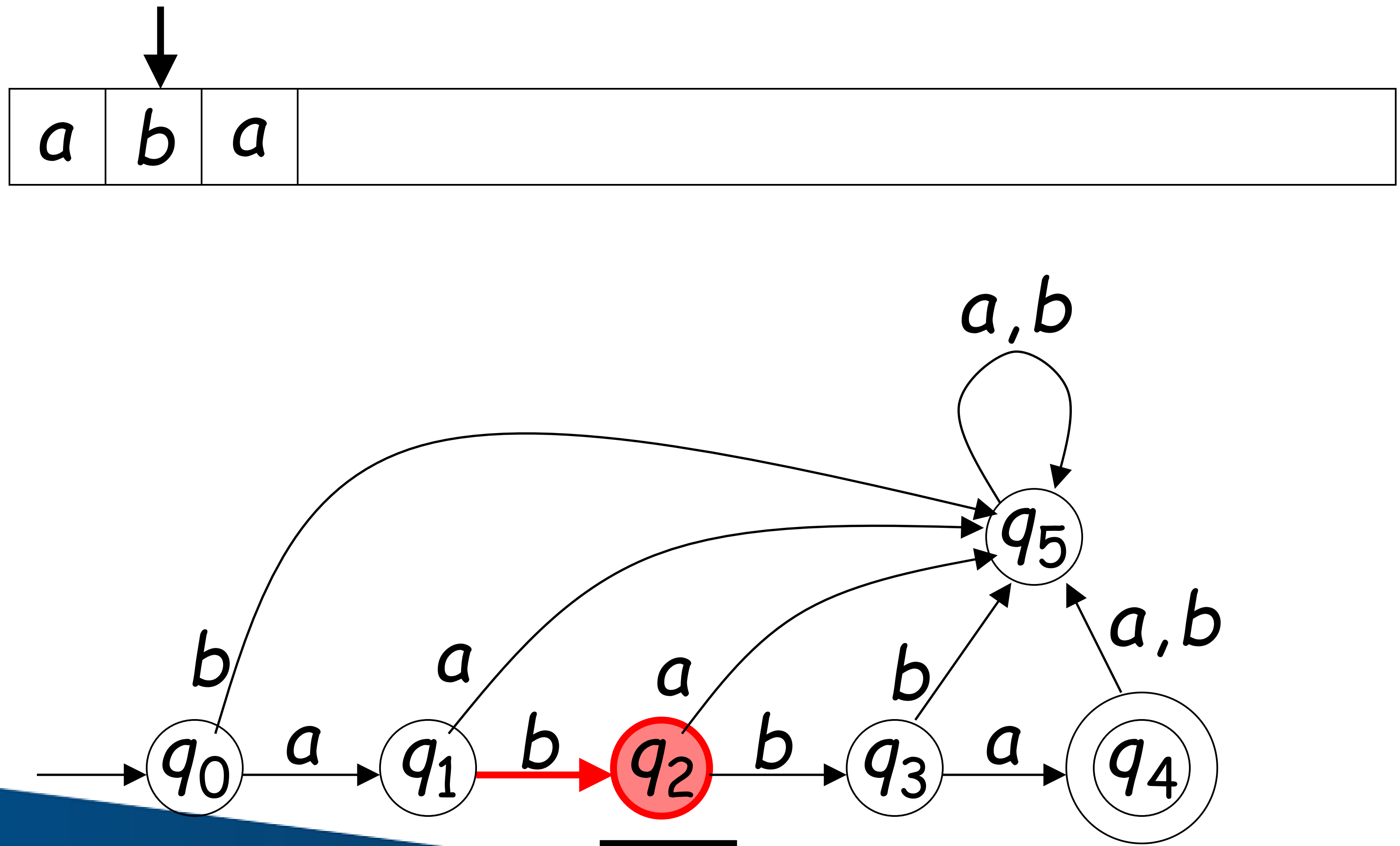
Rejection

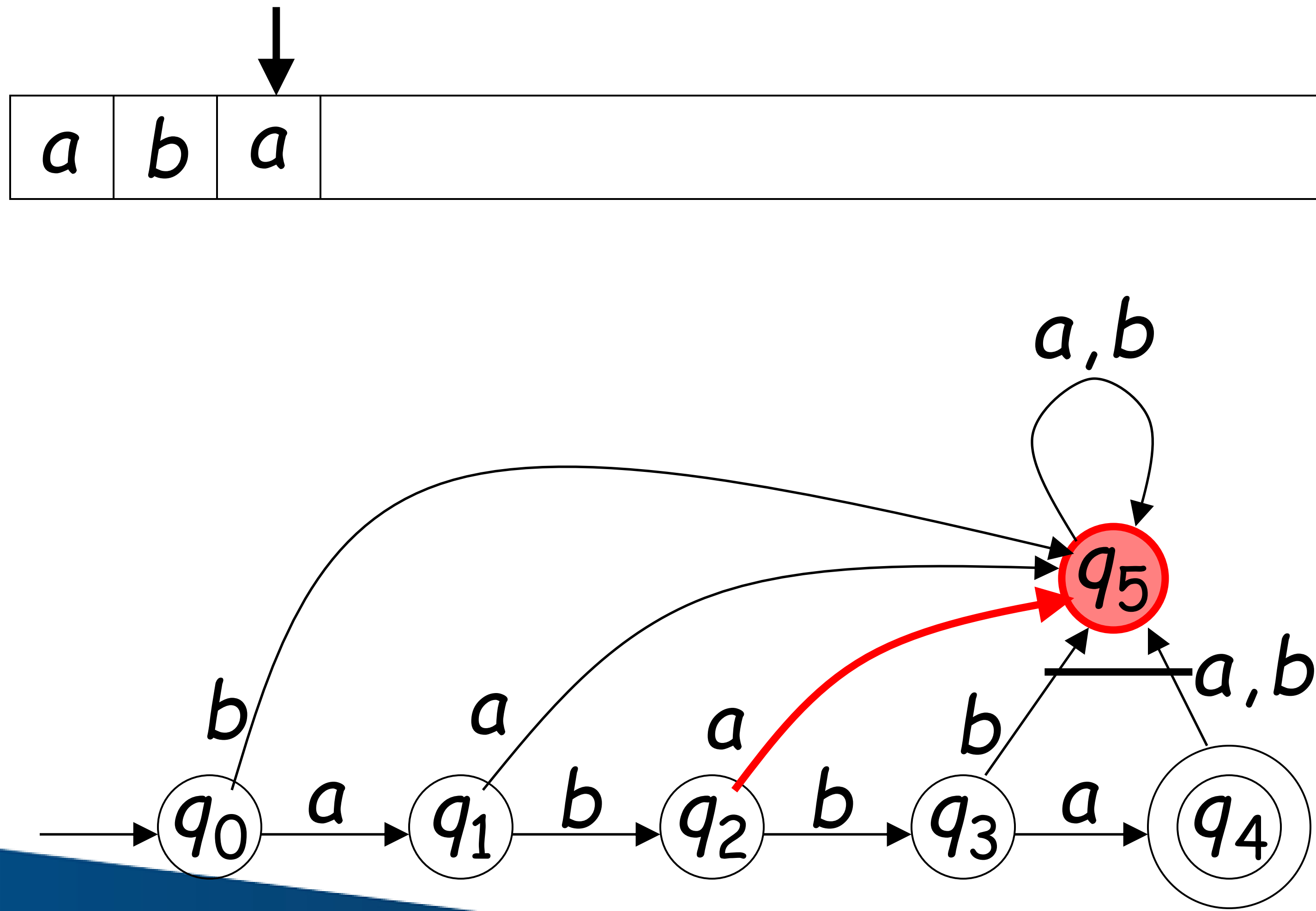
a	b	a	



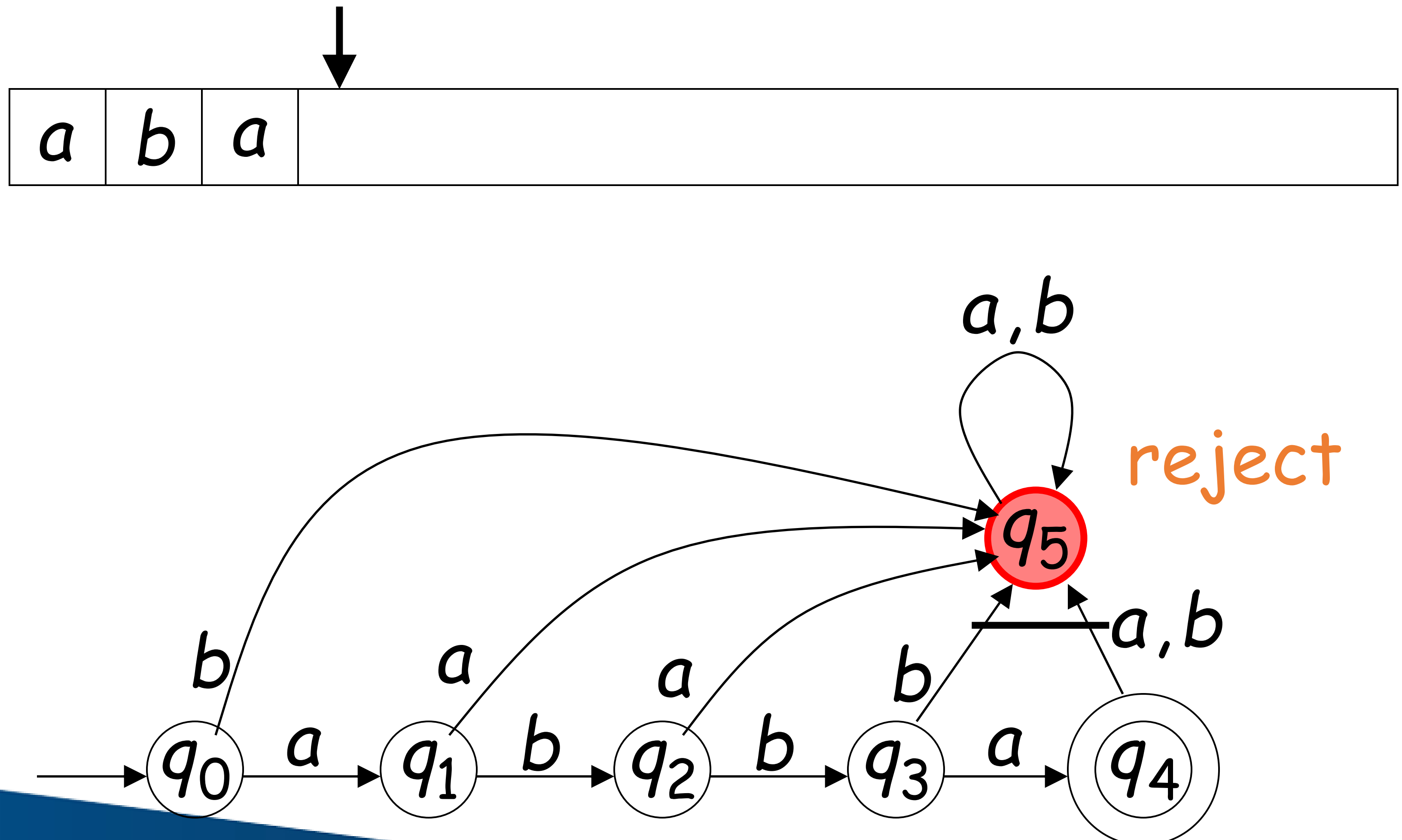


a
b
a
a,b
b
a
a
b
a
b
b
a
a,b
q_0
q_1
q_2
q_3
q_4
q_5


a
b
a
a,b
b
a
a
b
a
b
b
a
a,b
q0
q1
q2
q3
q4
q5

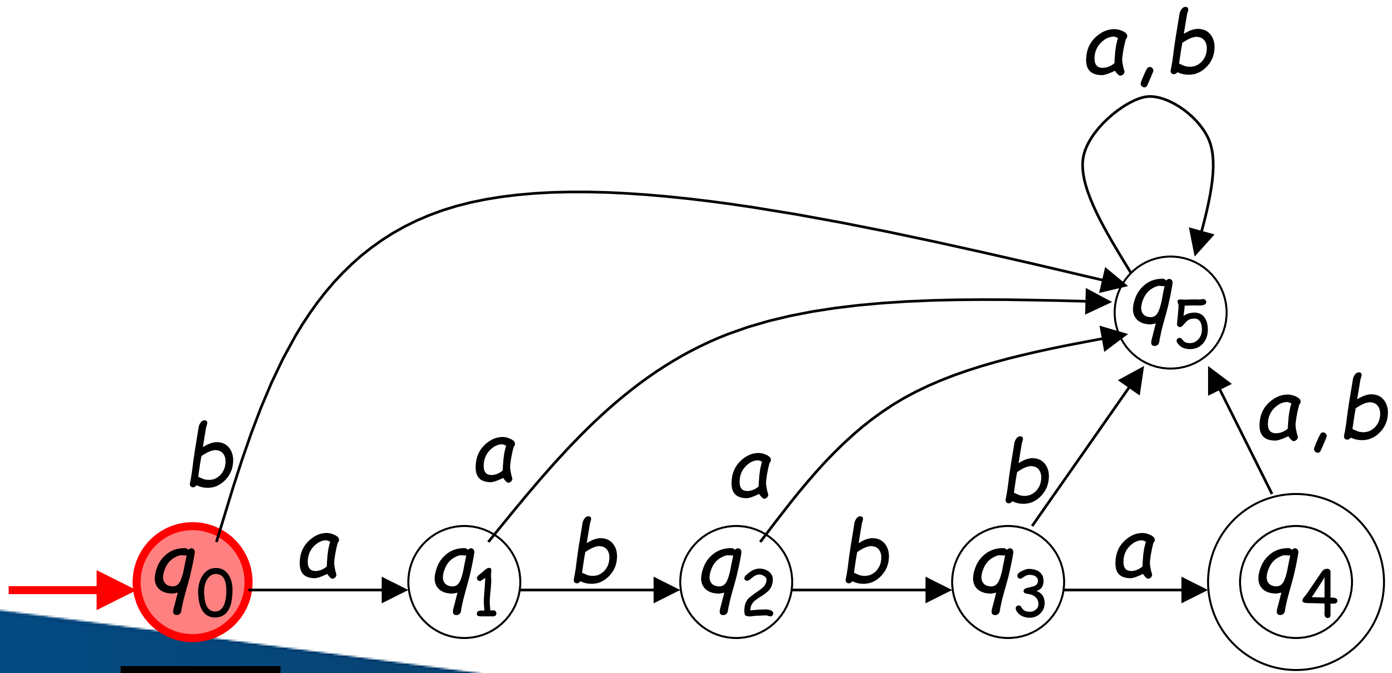

a
b
a
a,b
q5
b
a
a
b
a,b
q0
a
q1
b
q2
b
q3
a
q4

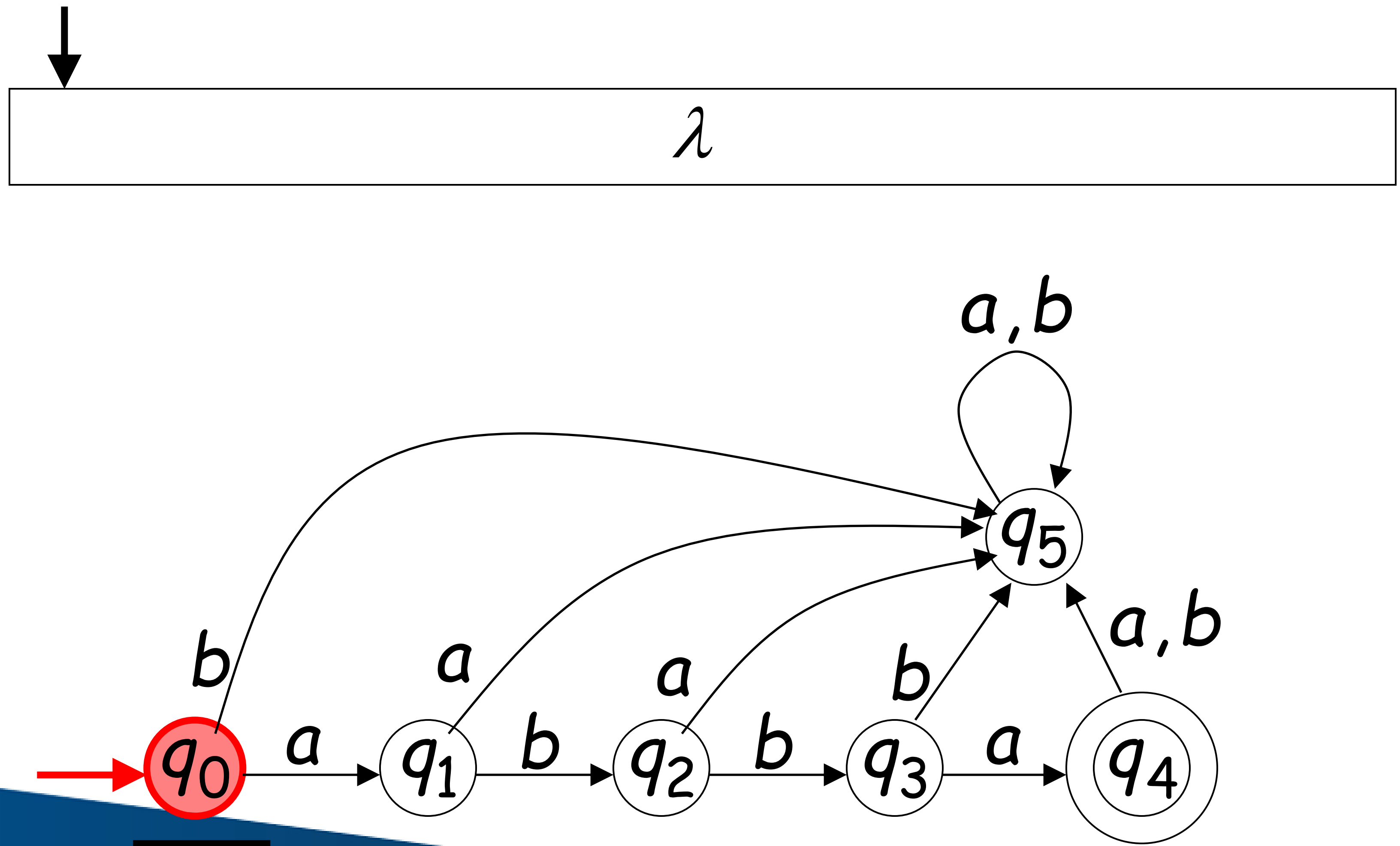
Input finished



Another Rejection

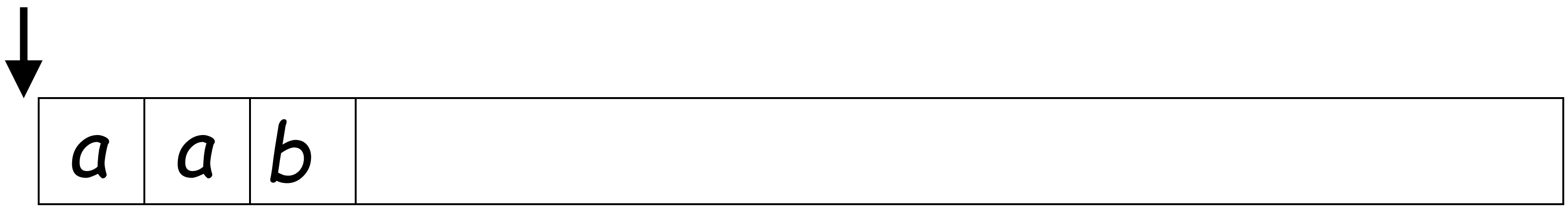
λ

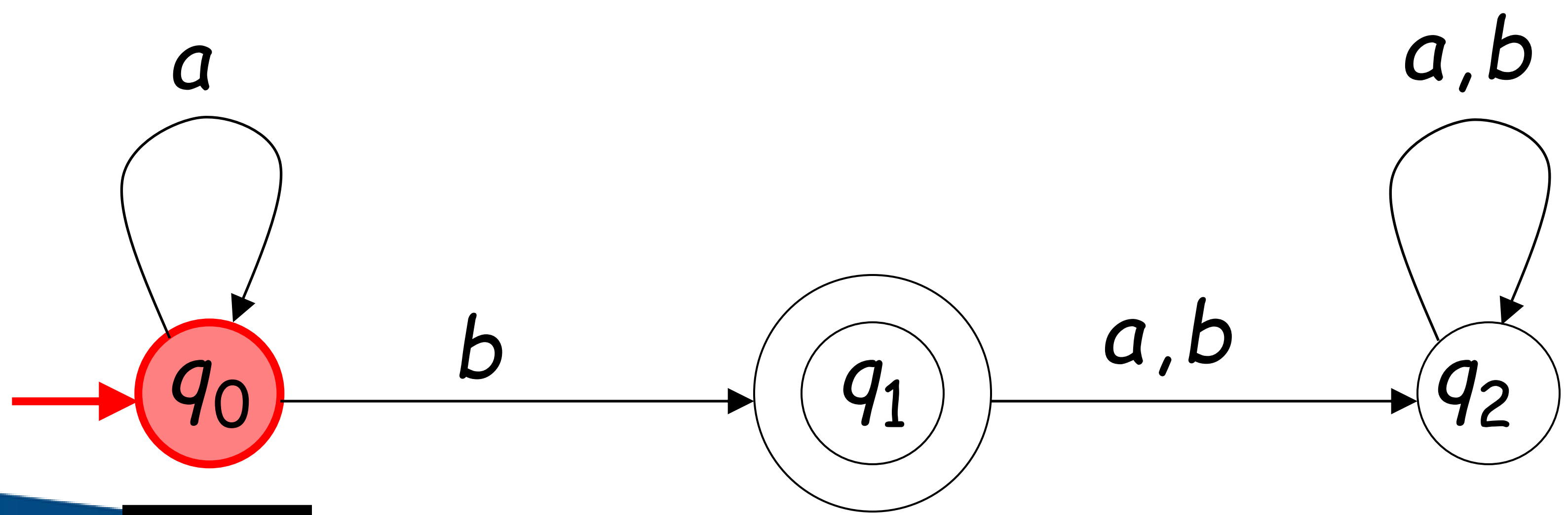


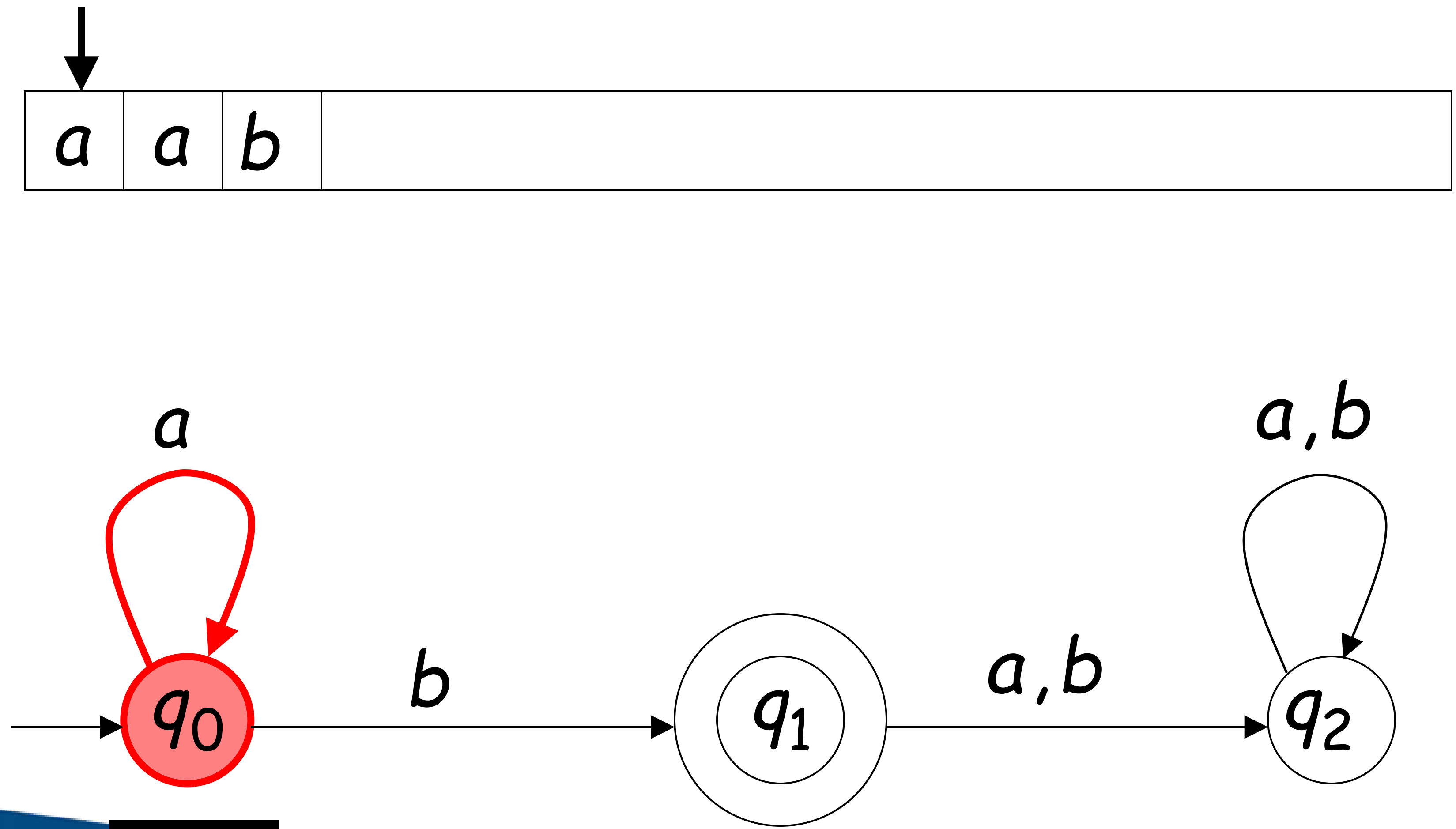


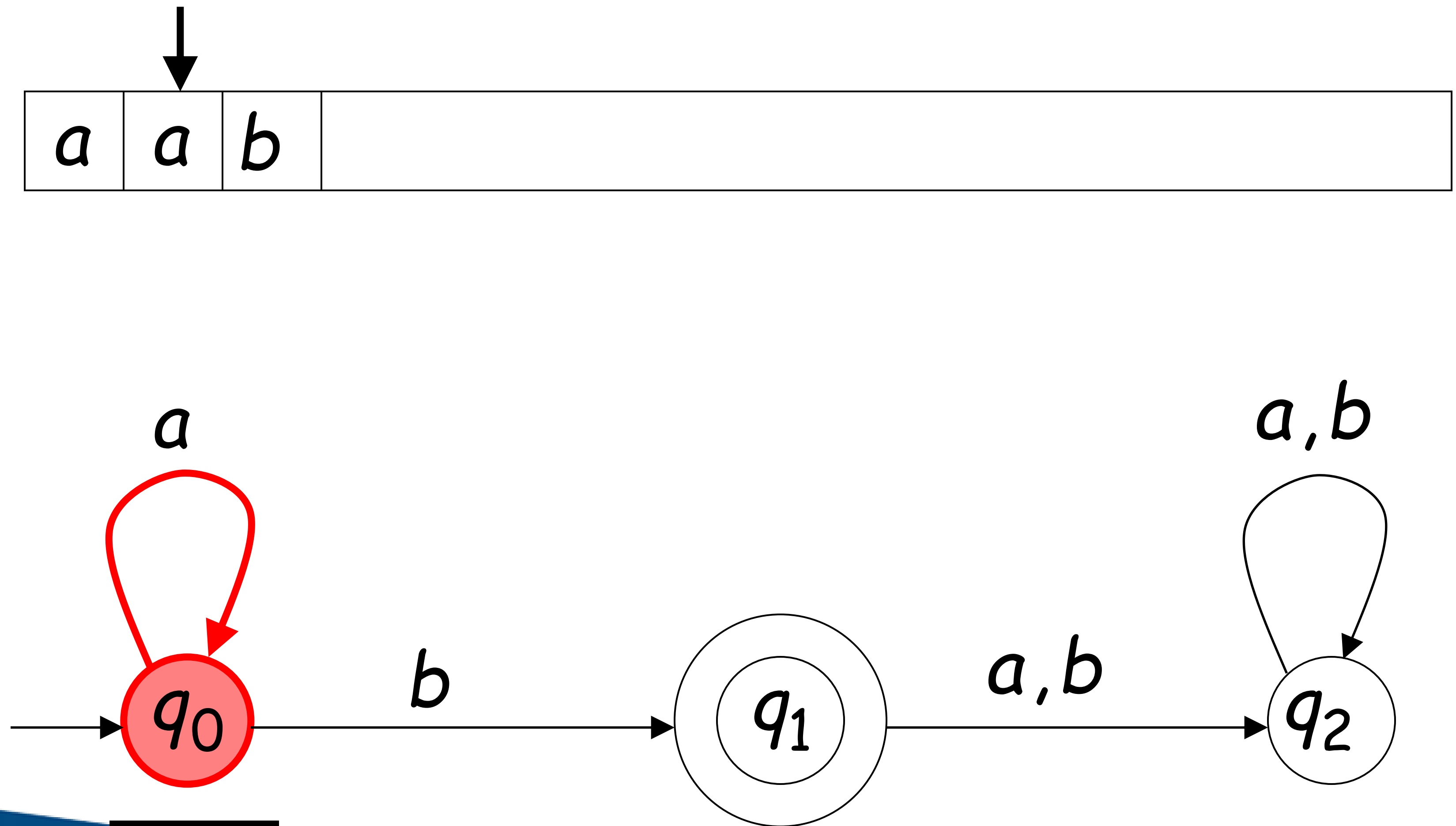
reject

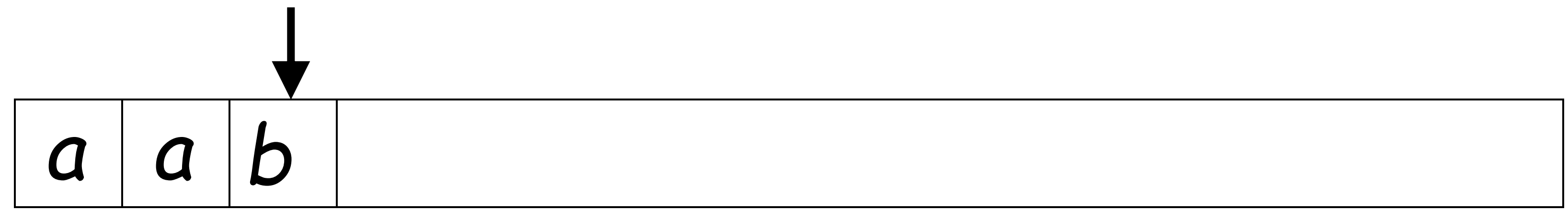
Another Example

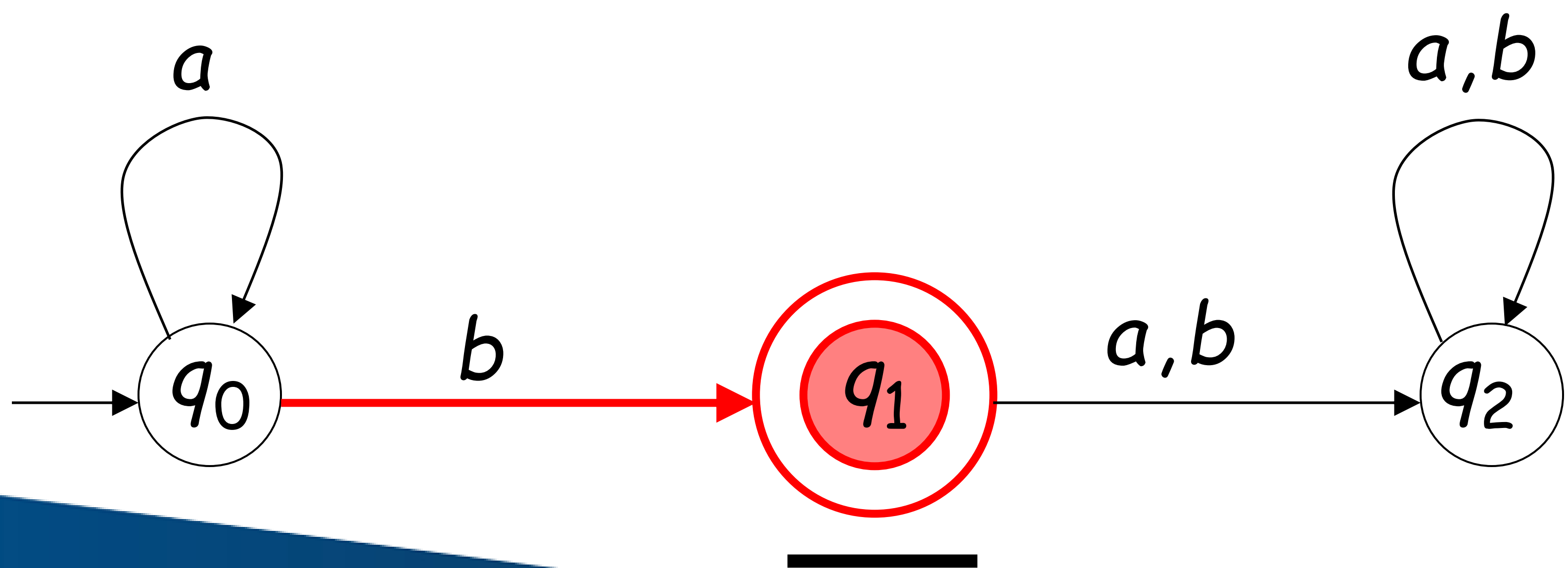




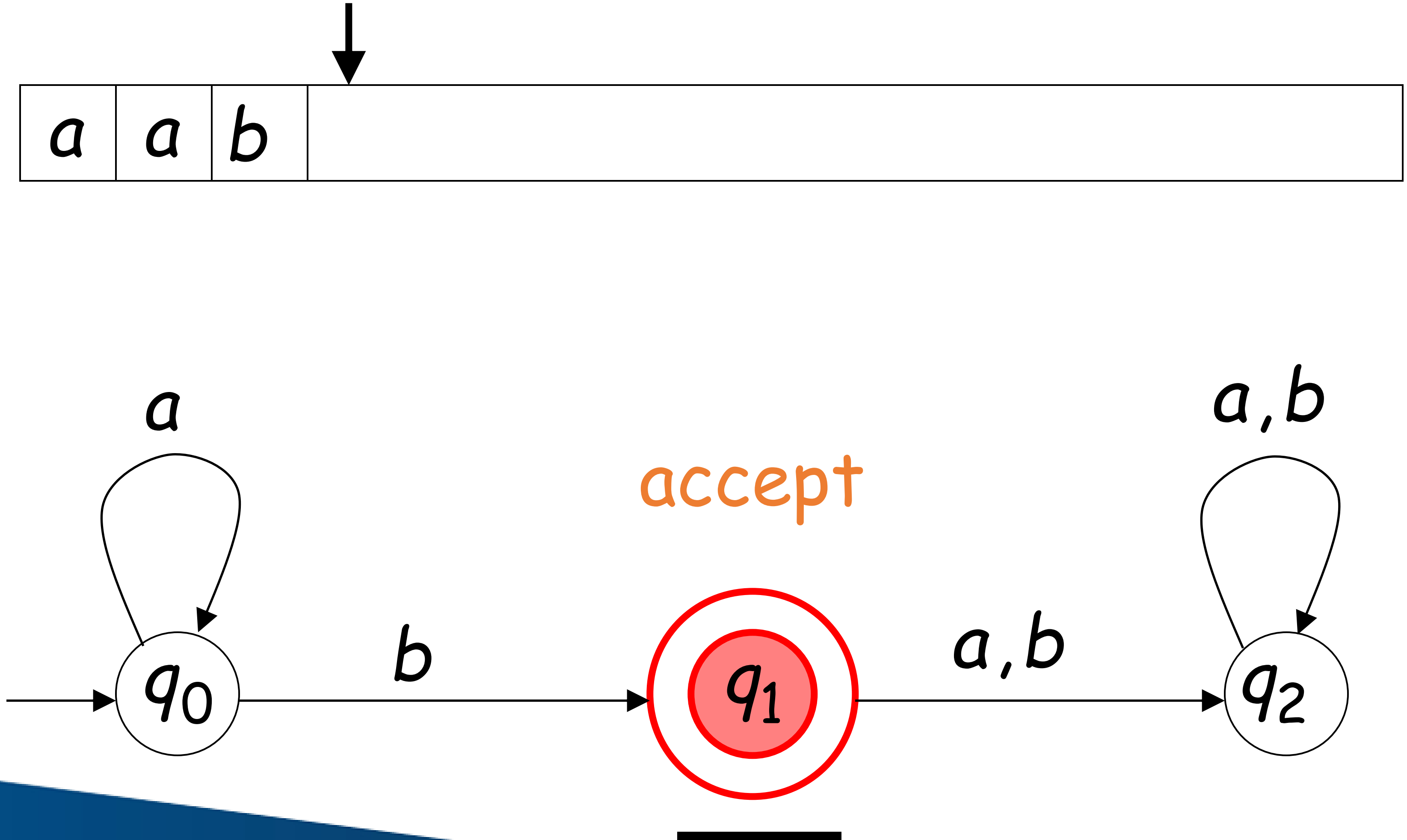

a
a
b
a
a,b
b
a,b
q_0
q_1
q_2


a
a
b
a
a,b
b
a,b
q_0
q_1
q_2

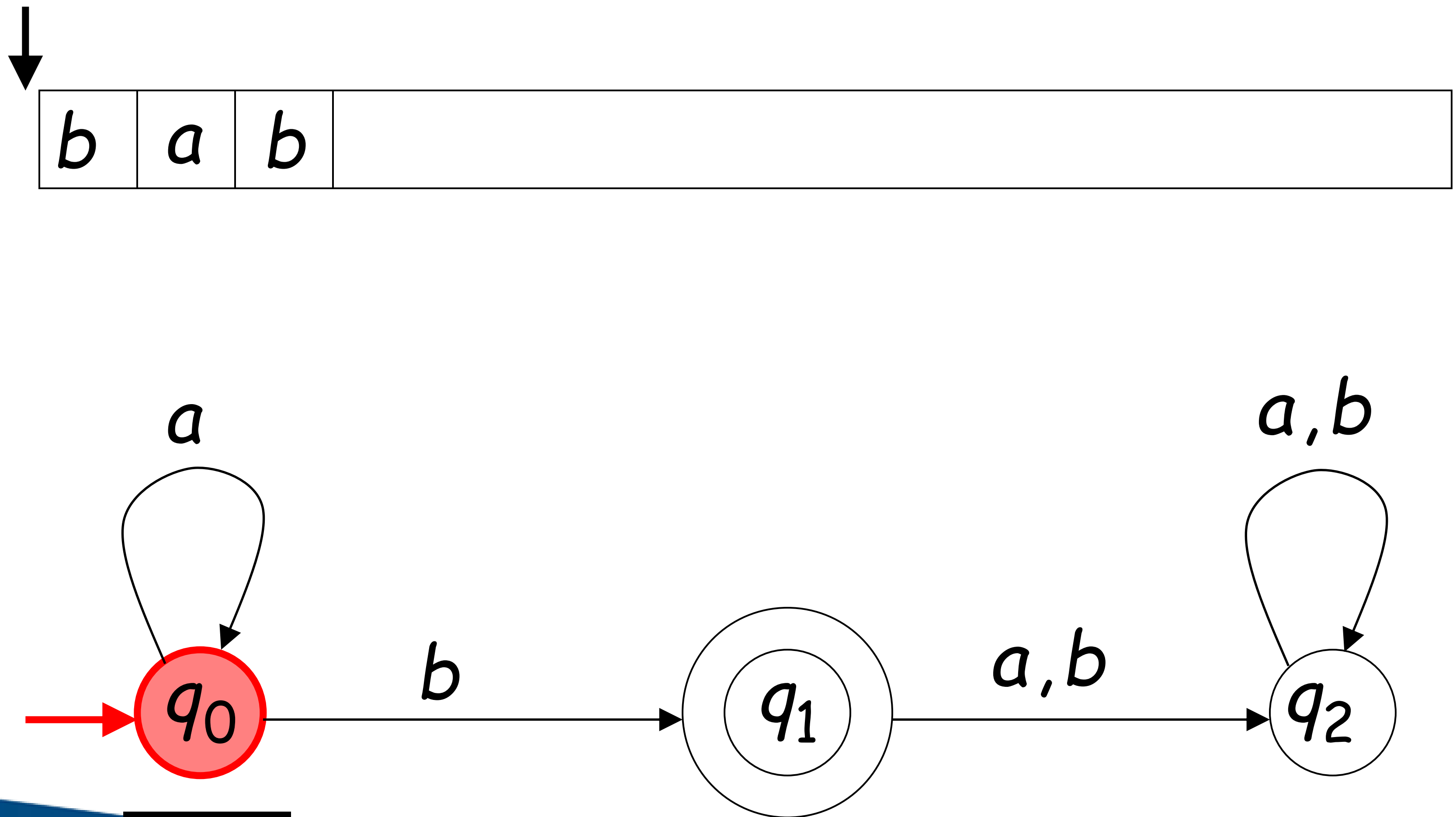

a
a
b

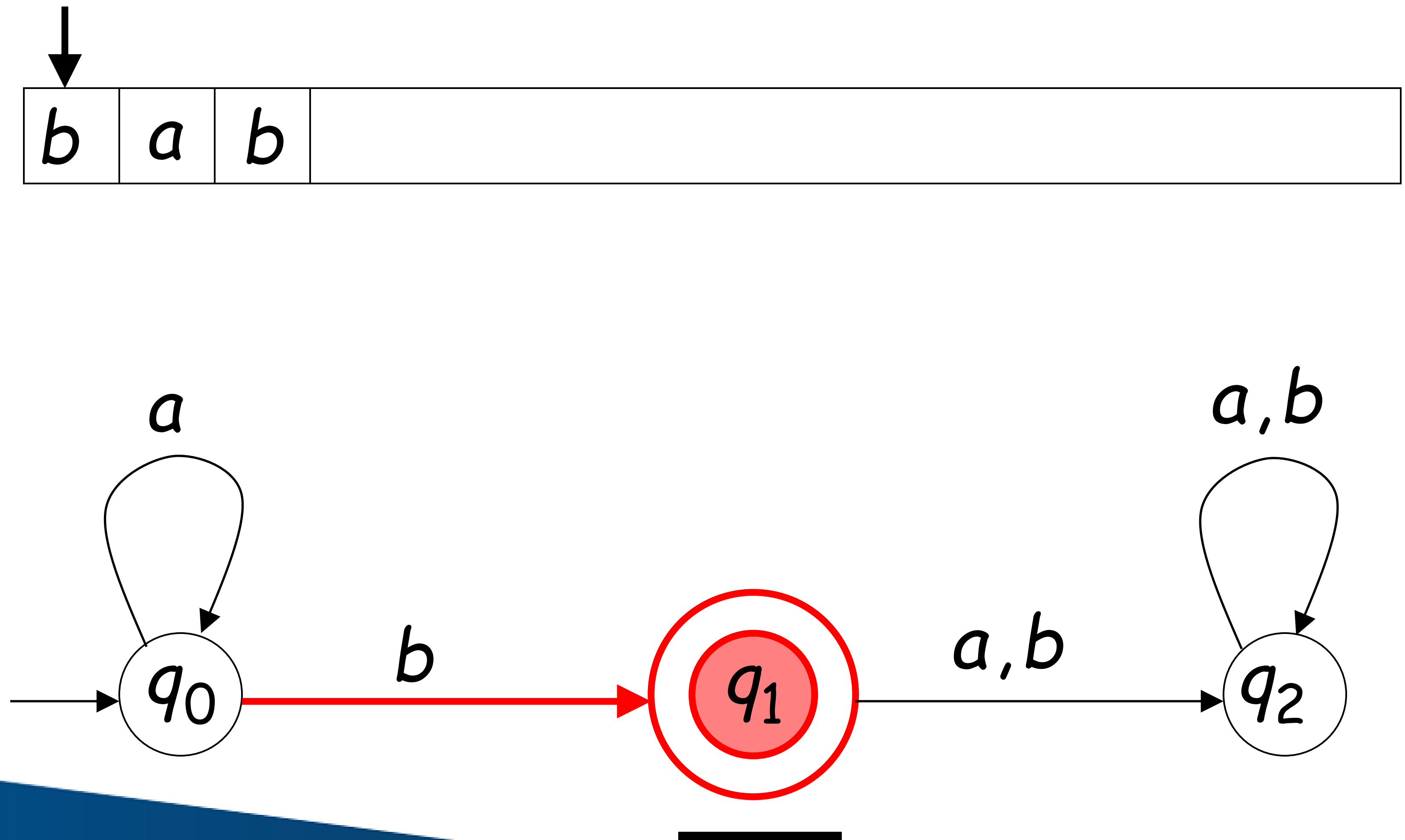

a
a,b
b
a,b
q0
q1
q2

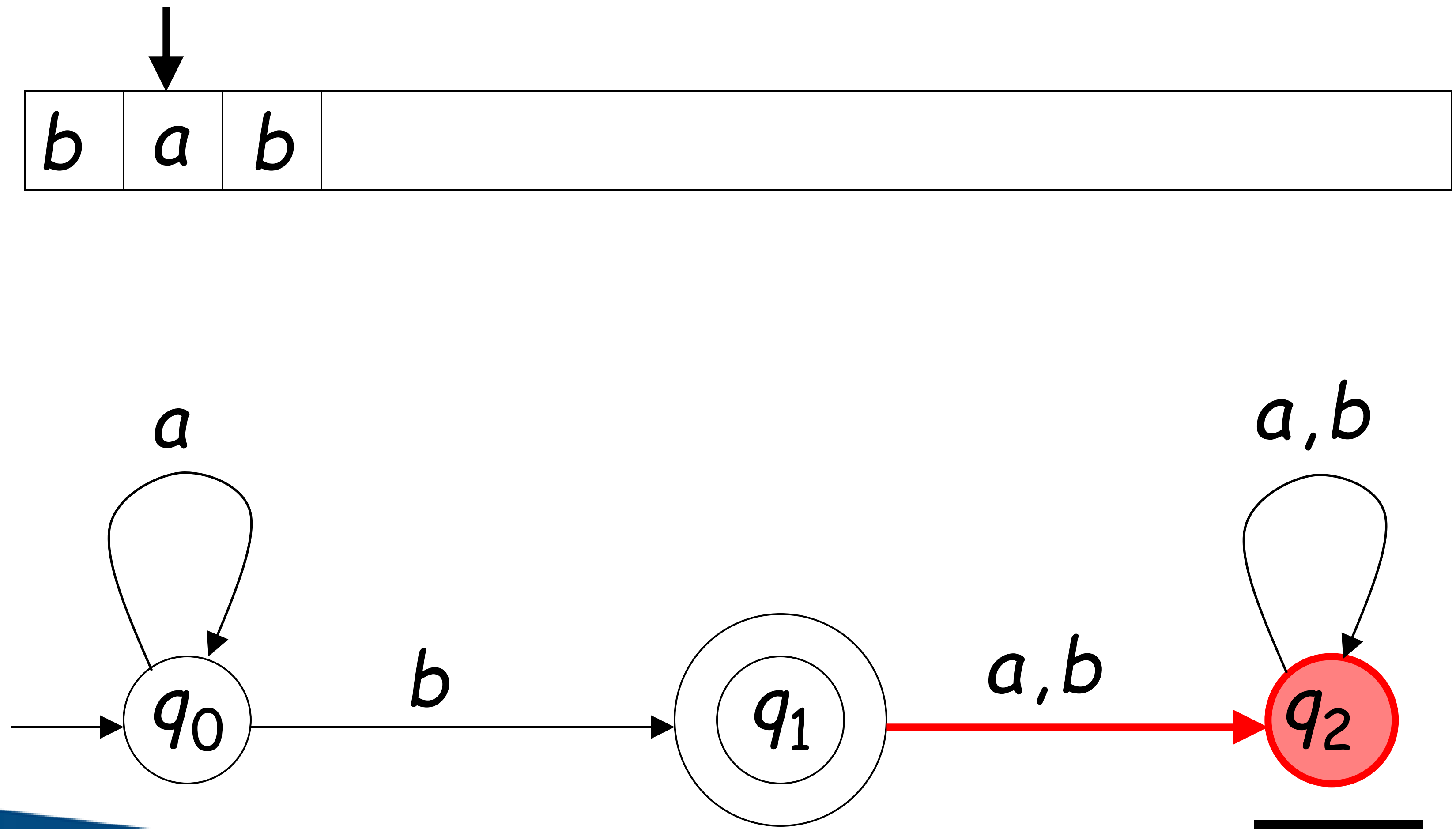
Input finished

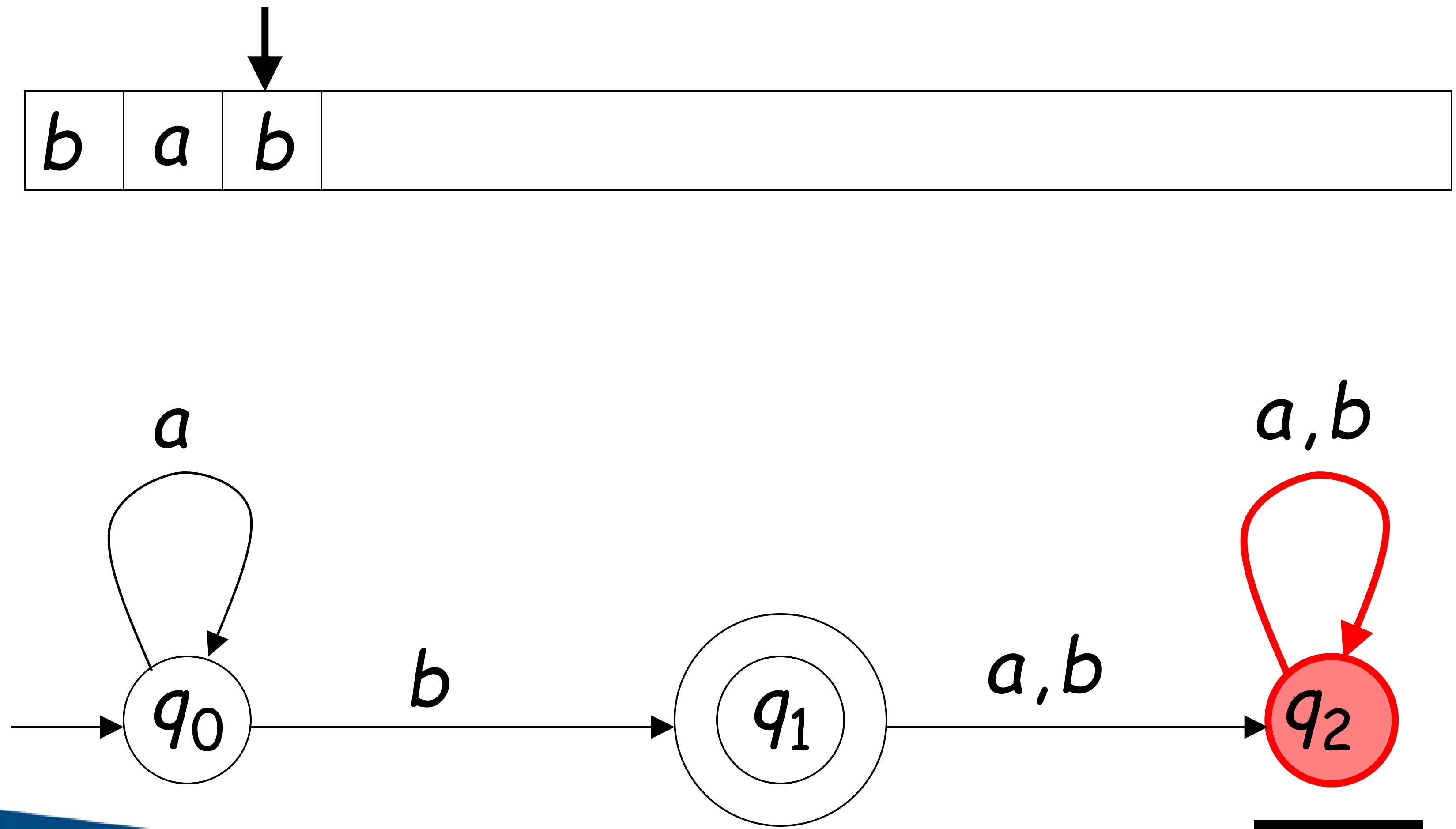


Rejection Example




b
a
b
a
a,b
b
a,b
q_0
q_1
q_2

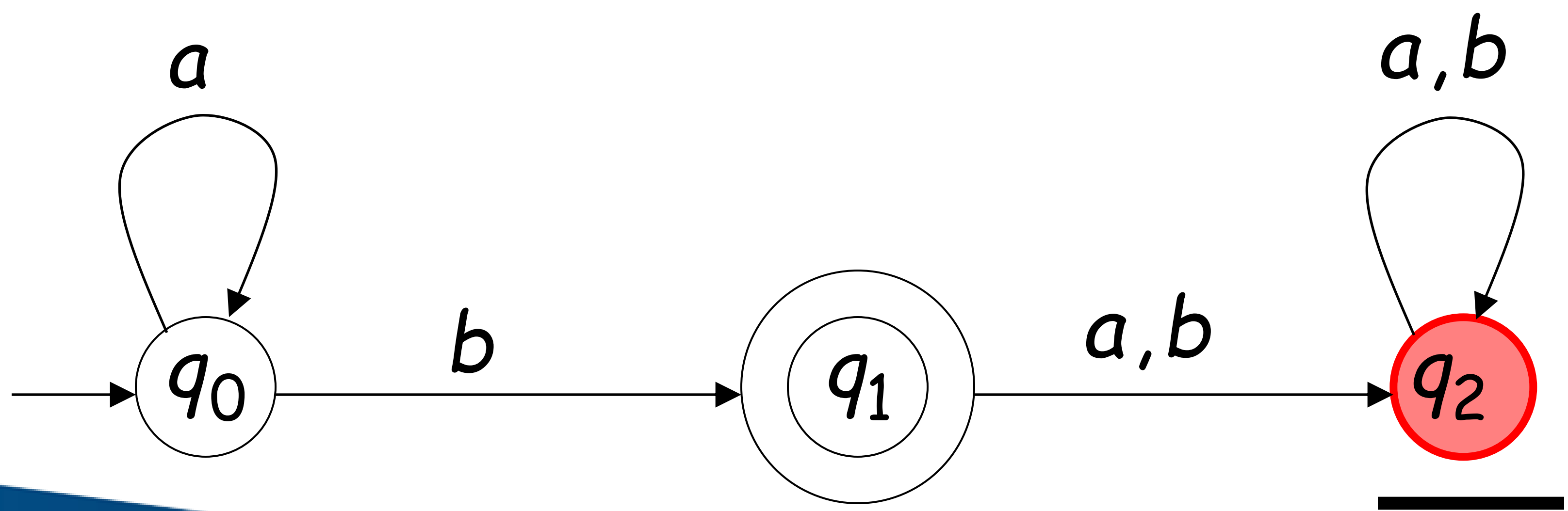

b
a
b
a
a,b
b
a,b
q_0
q_1
q_2



b
a
b
a
a,b
b
a,b
q_0
q_1
q_2

Input finished

b	a	b	



reject

Languages Accepted by FAs

Definition:

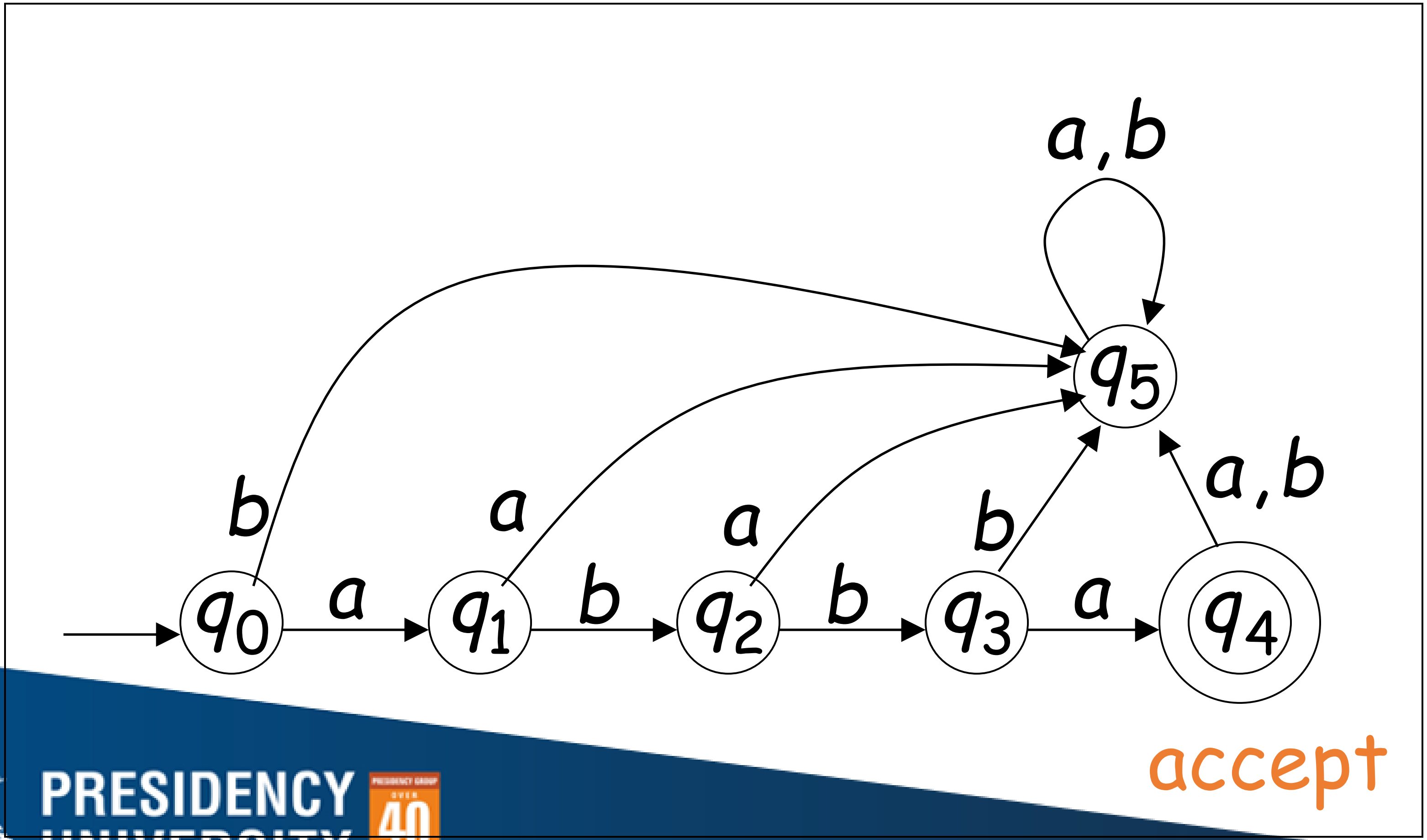
The language $L(M)$ contains all input strings accepted by M

$L(M)$ = { strings that bring M to an accepting state}

Example

$$L(M) = \{abba\}$$

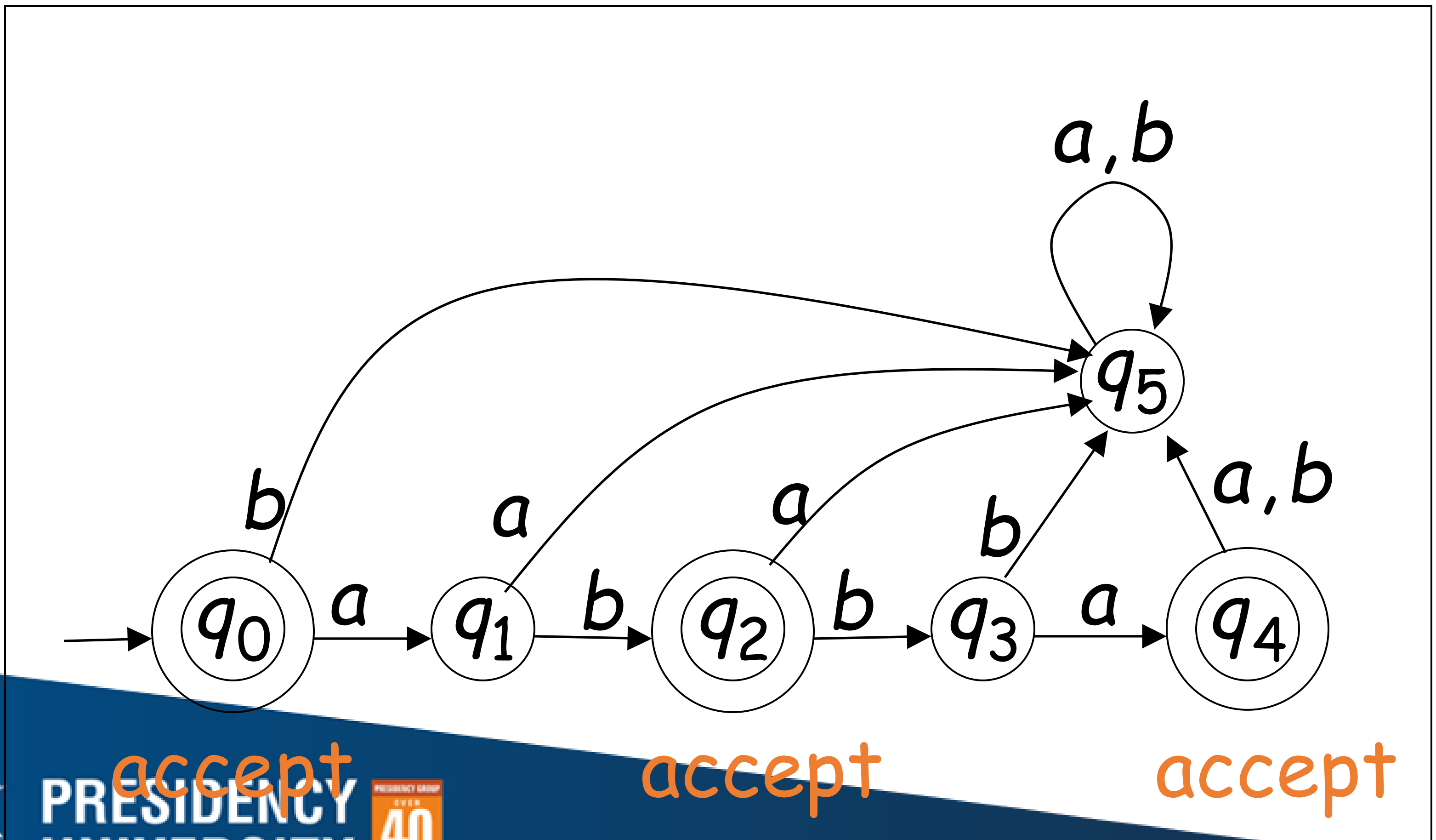
M



Example

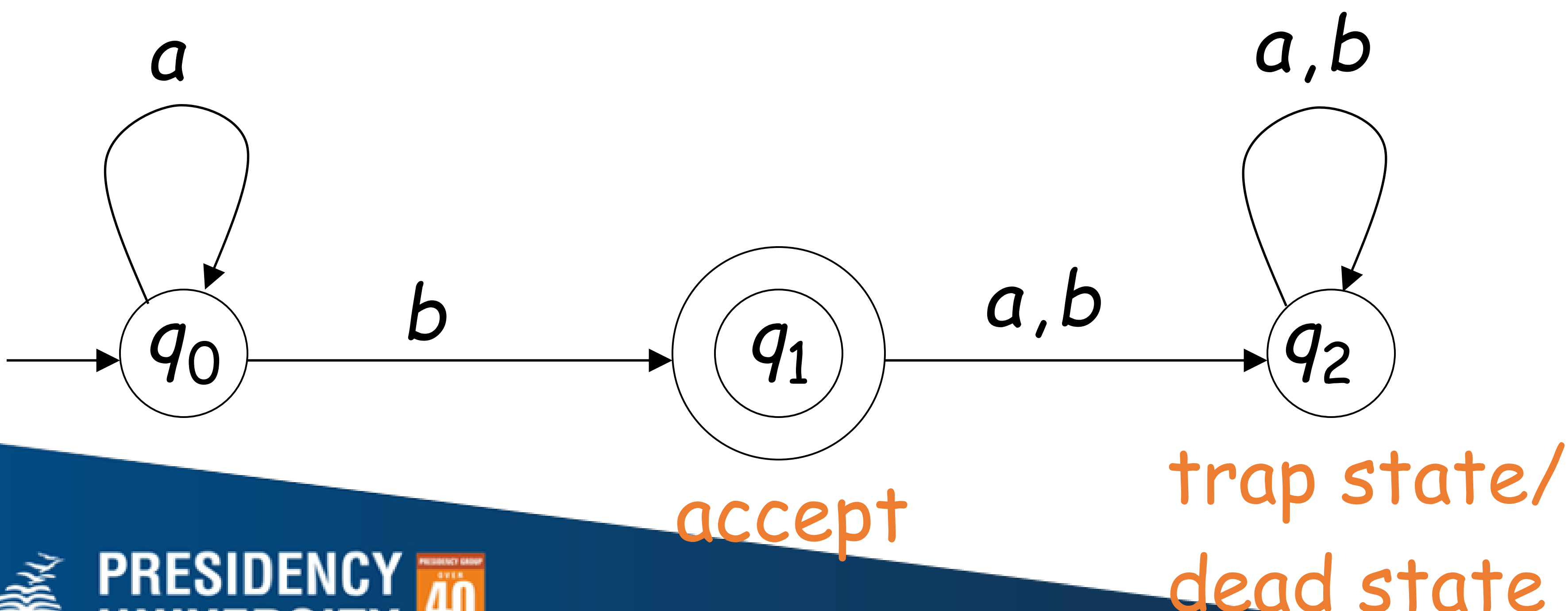
$$L(M) = \{\lambda, ab, abba\}$$

M



Example

$$L(M) = \{a^n b : n \geq 0\}$$



Formal Definition

- Finite Automaton (FA)

$$M = (Q, \Sigma, \delta, q_0, F)$$

Q : set of states

Σ : input alphabet

δ : transition function

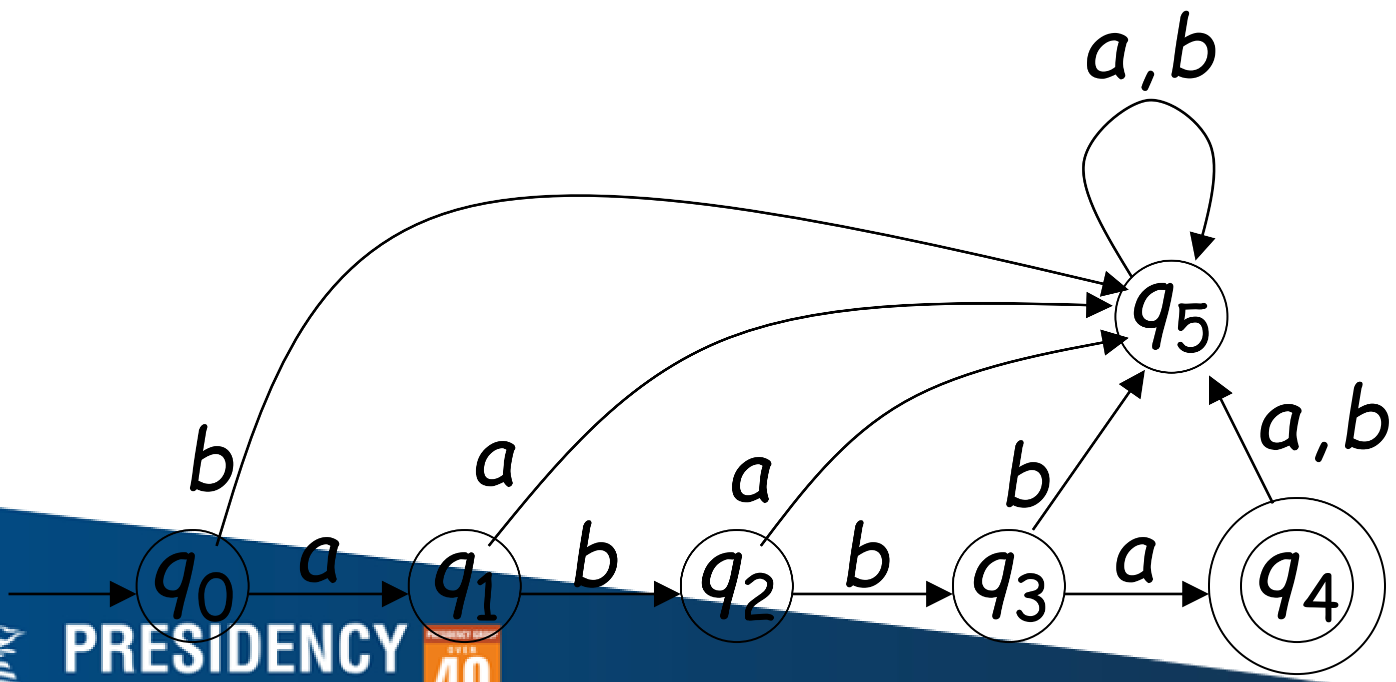
q_0 : initial state

F : set of accepting states

PRESIDENCY UNIVERSITY
GAIN MORE KNOWLEDGE REACH GREATER HEIGHTS
Private University Estd. in Karnataka State by Act No. 41 of 2013

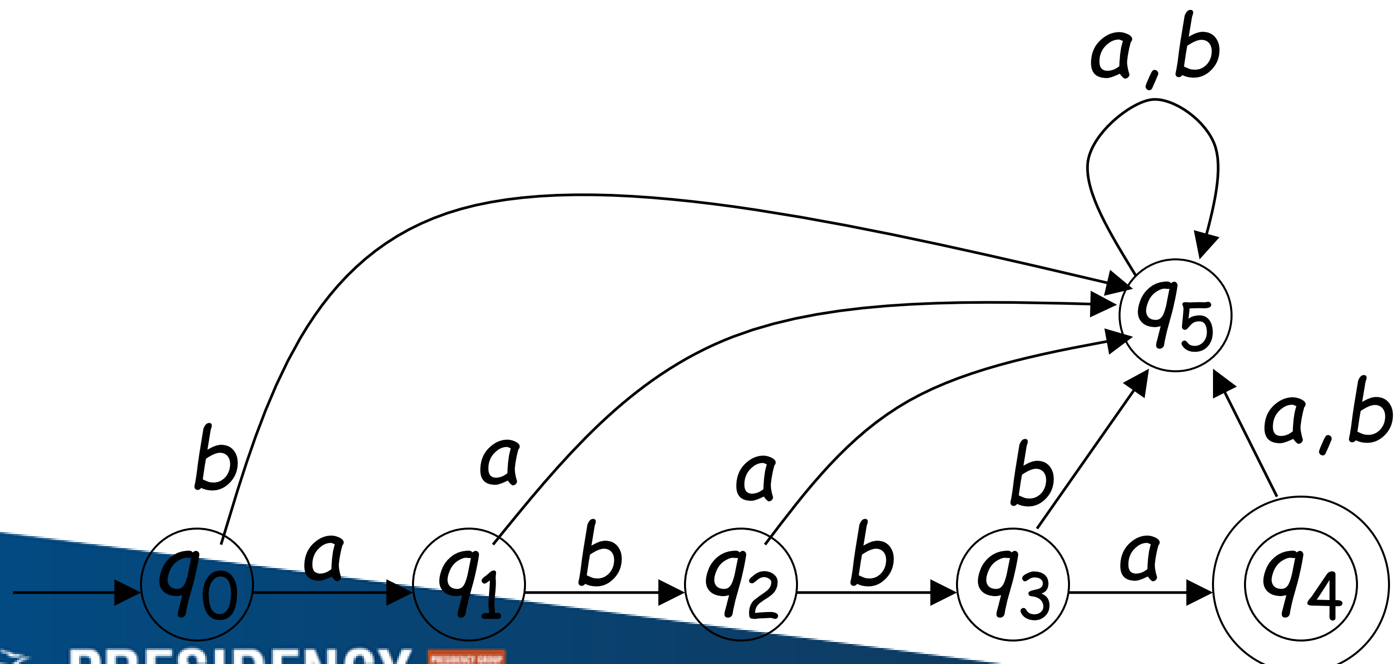
Input Alphabet Σ

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



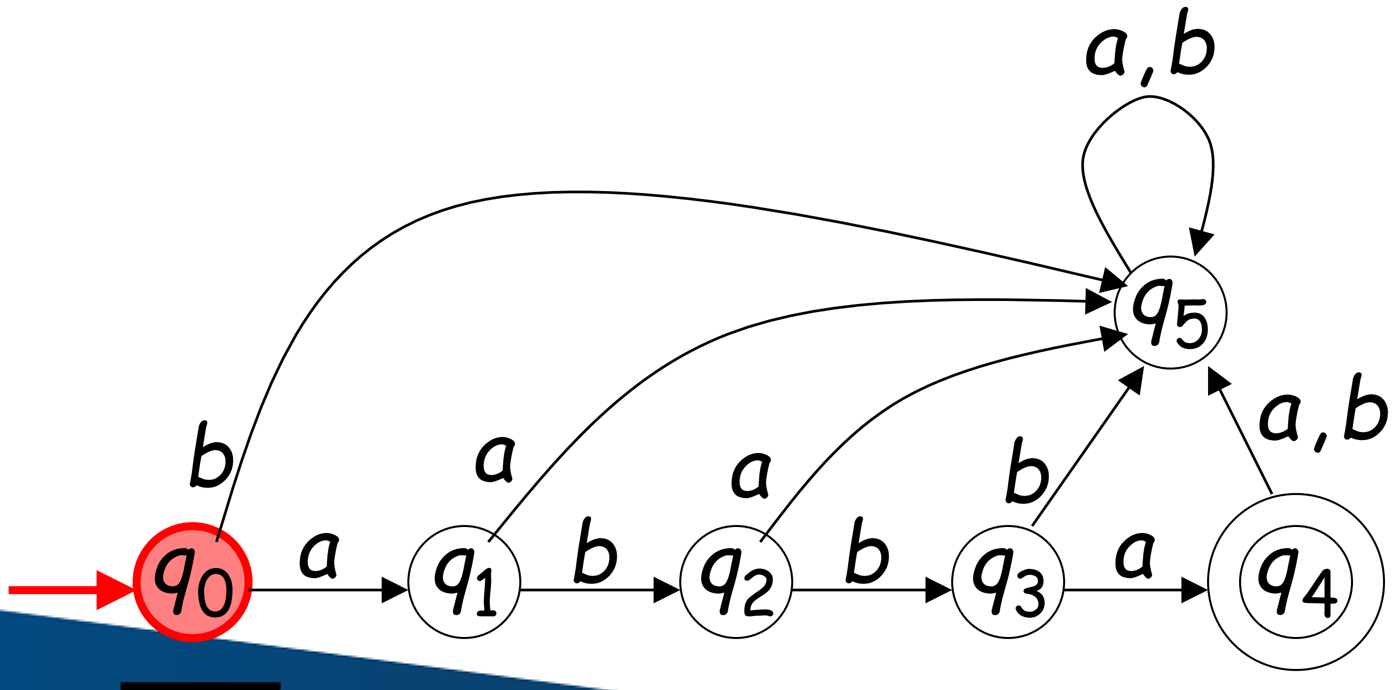
Set of States Q

$$Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$$



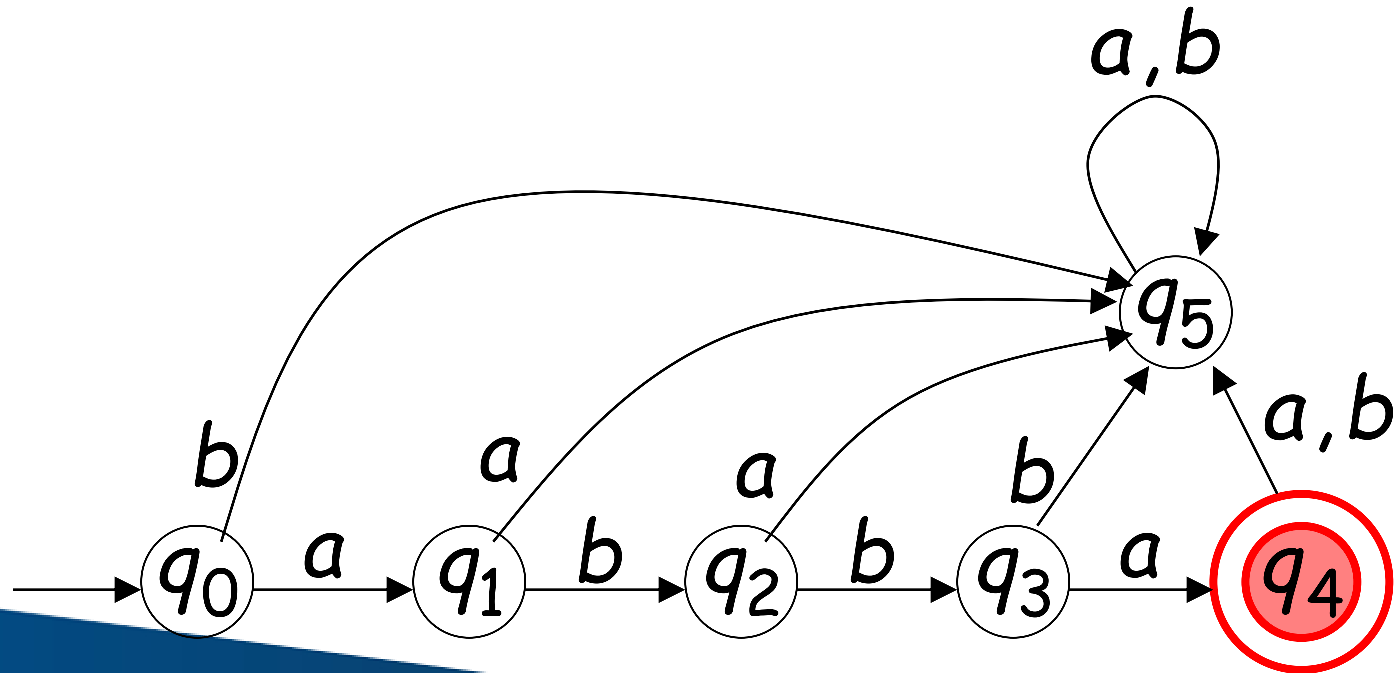
Initial State q_0

•



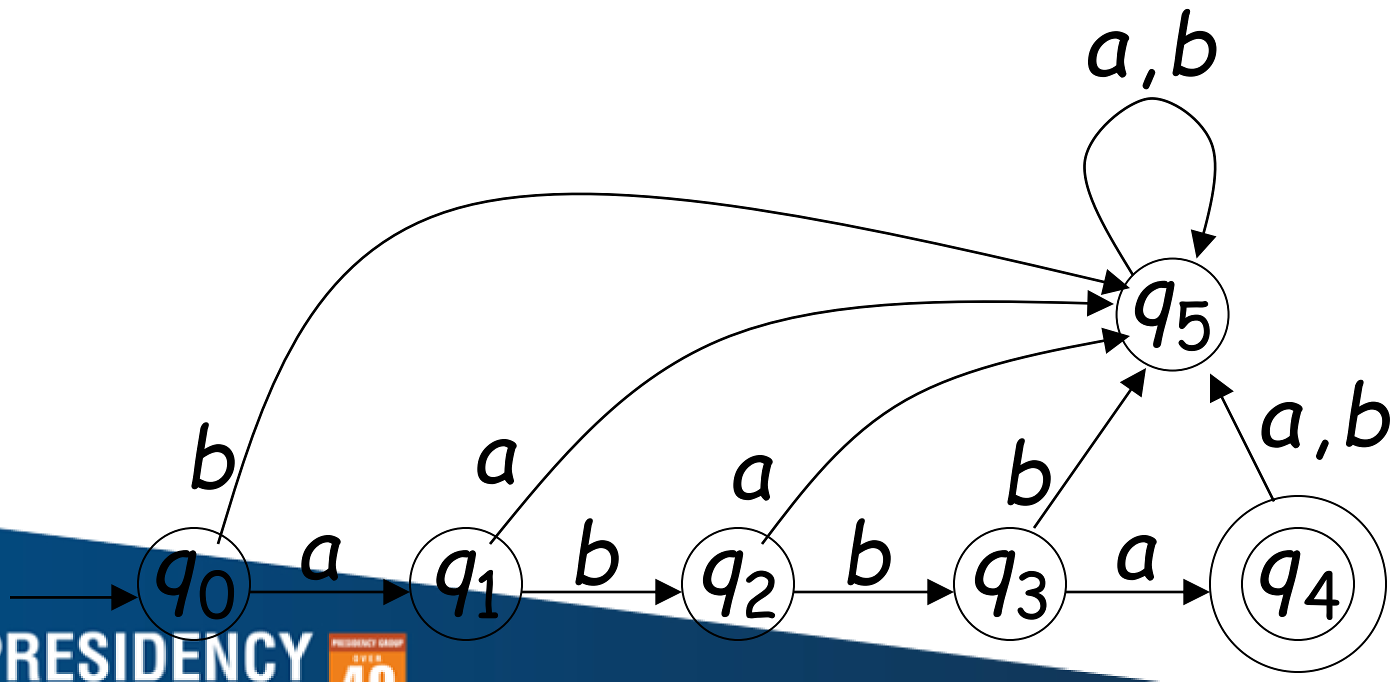
Set of Accepting States F

- $F = \{q_4\}$

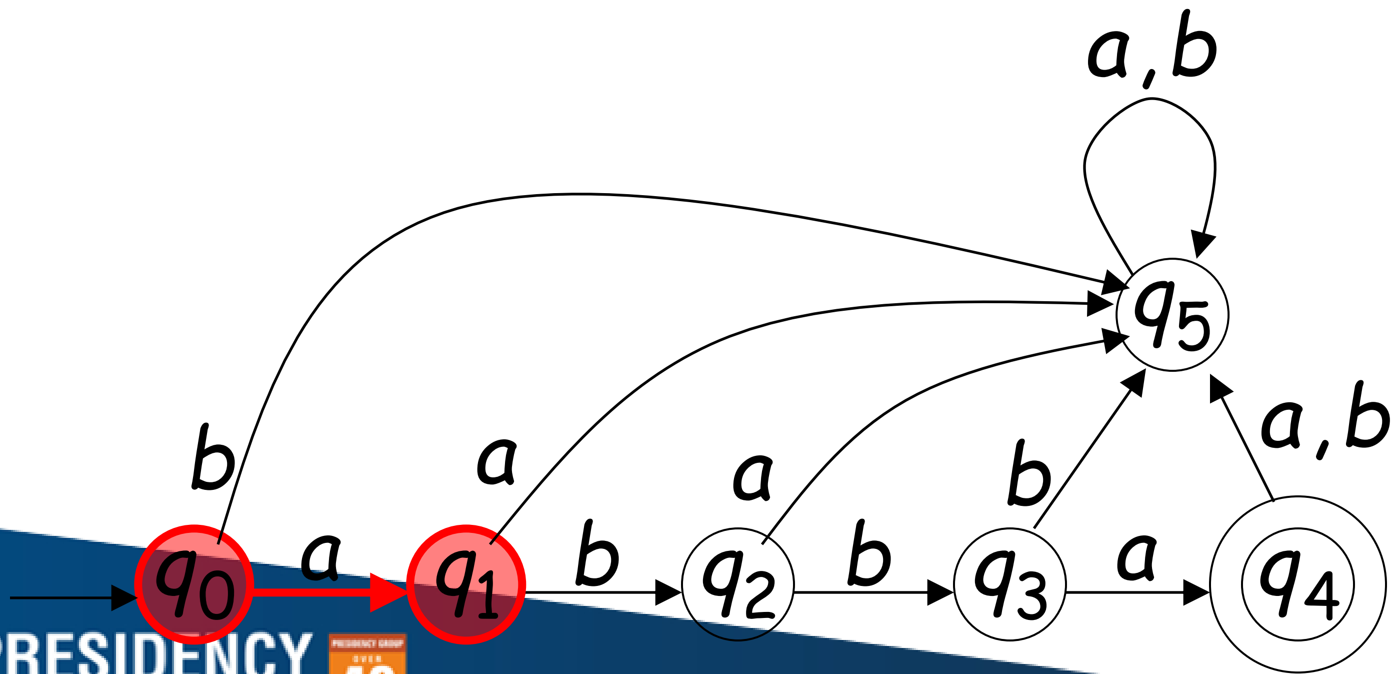


Transition Function δ

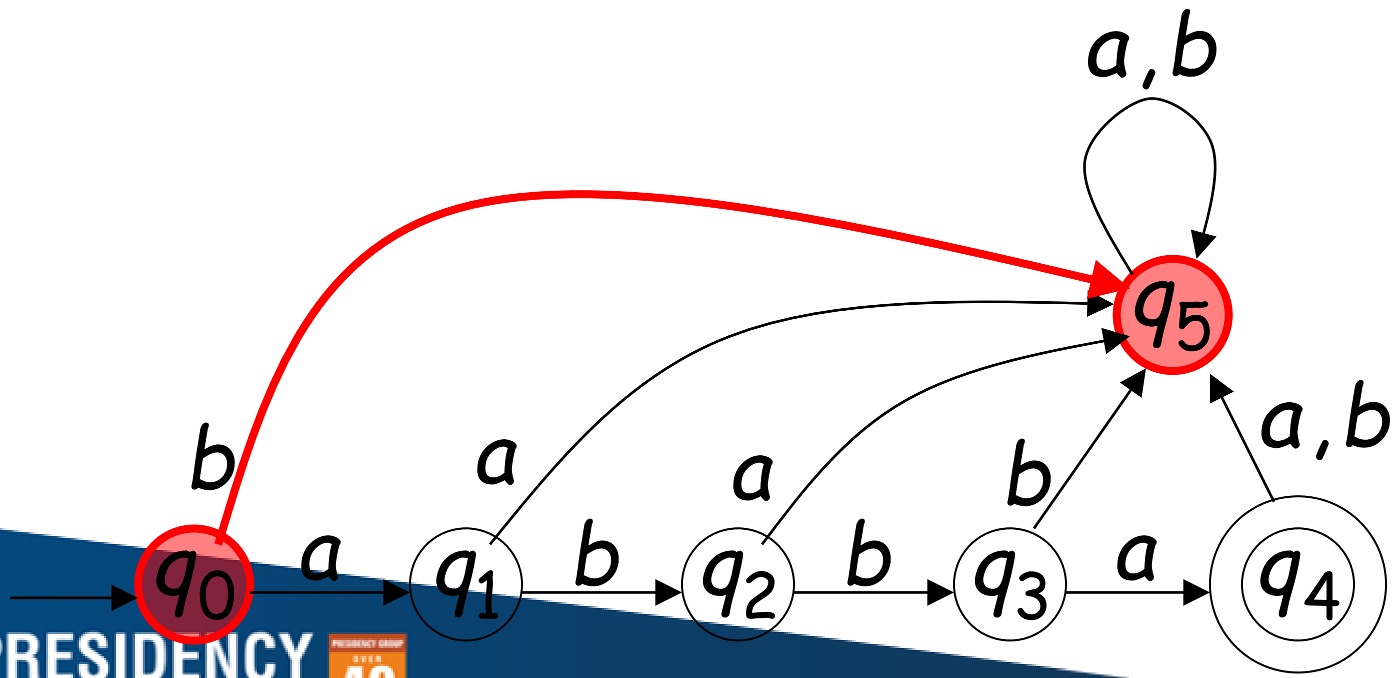
- $$\delta : Q \times \Sigma \rightarrow Q$$



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



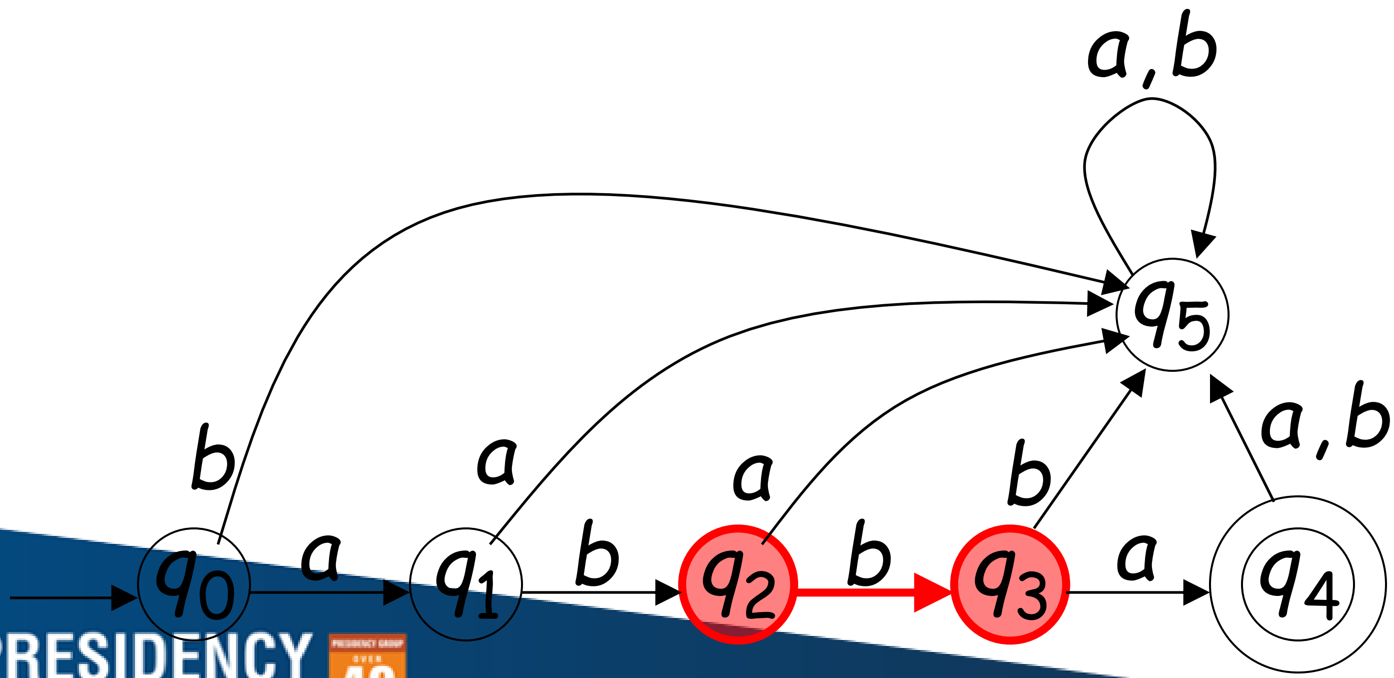
$$\delta(q_0, b) = q_5$$



PRESIDENCY UNIVERSITY

Private University Estd. in Karnataka State by Act No. 41 of 2013

$$\delta(q_2, b) = q_3$$

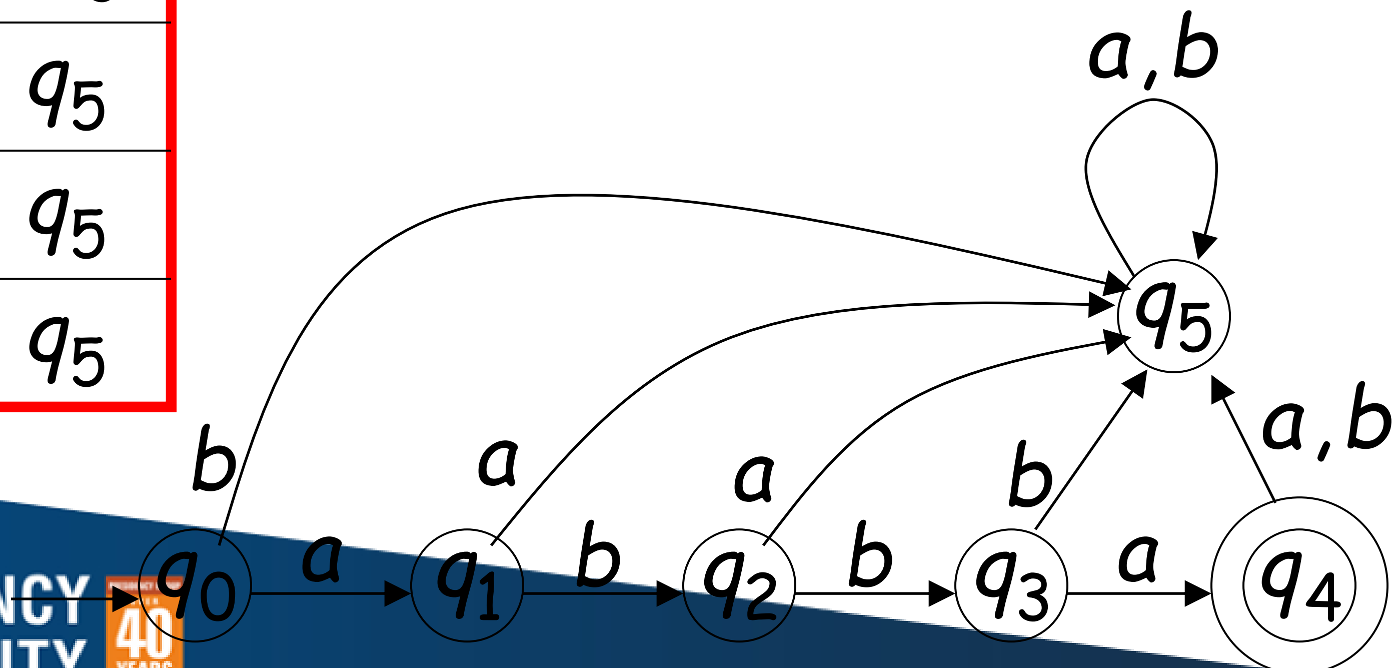


PRESIDENCY UNIVERSITY

Private University Estd. in Karnataka State by Act No. 41 of 2013

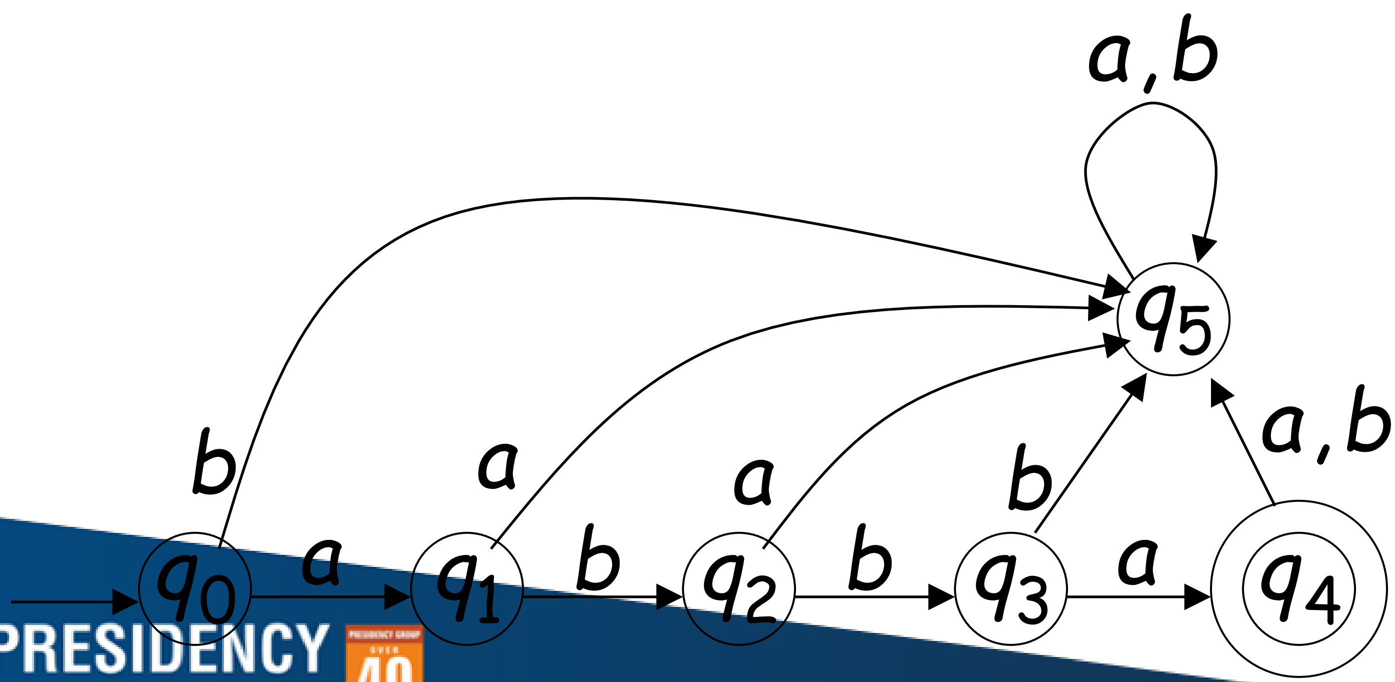
Transition Table δ

δ	a	b
$\rightarrow q_0$	q_1	q_5
q_1	q_5	q_2
q_2	q_5	q_3
q_3	q_4	q_5
q_4	q_5	q_5
q_5 *	q_5	q_5

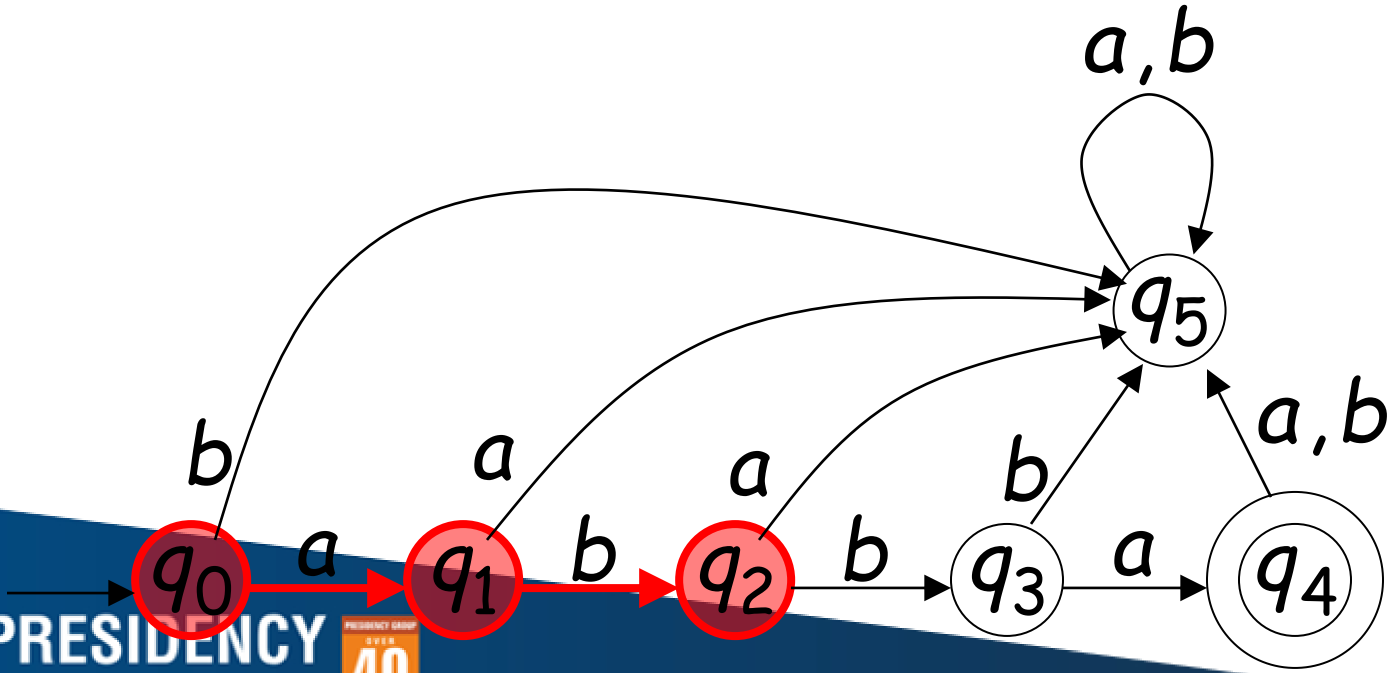


Extended Transition Function δ^*

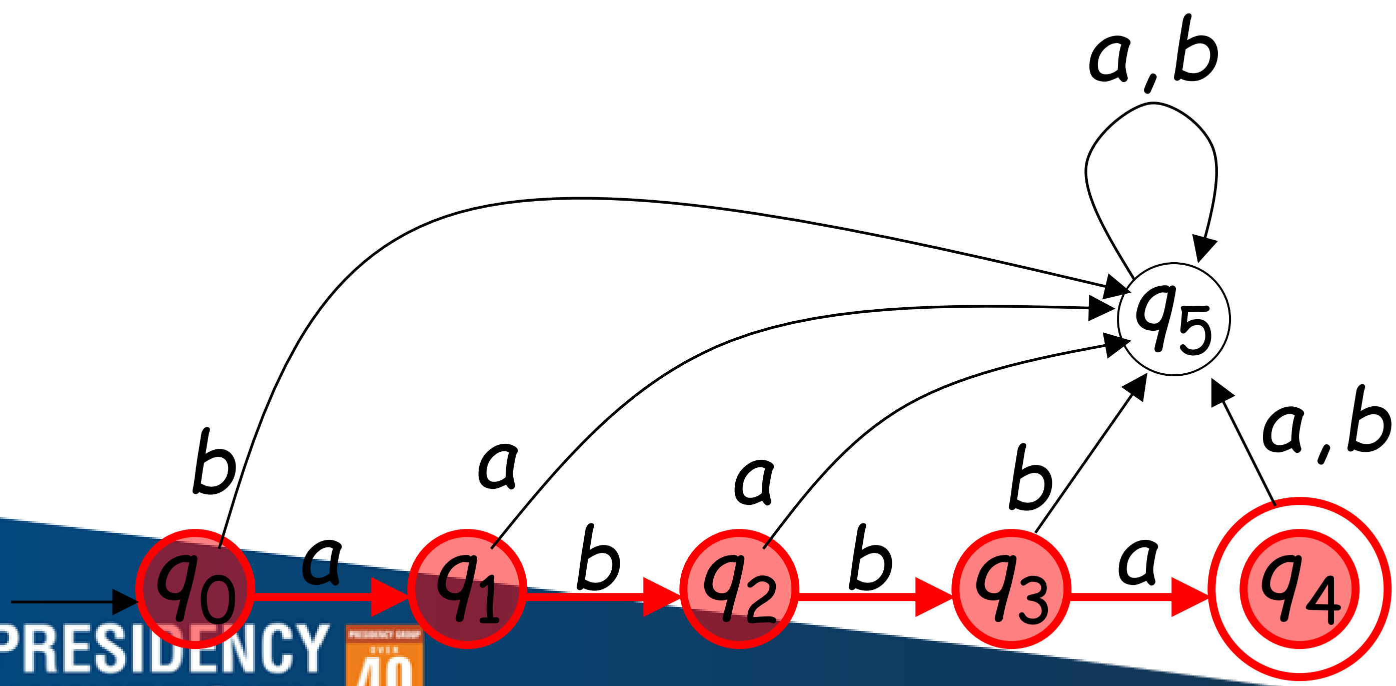
- $$\delta^* : Q \times \Sigma^* \rightarrow Q$$



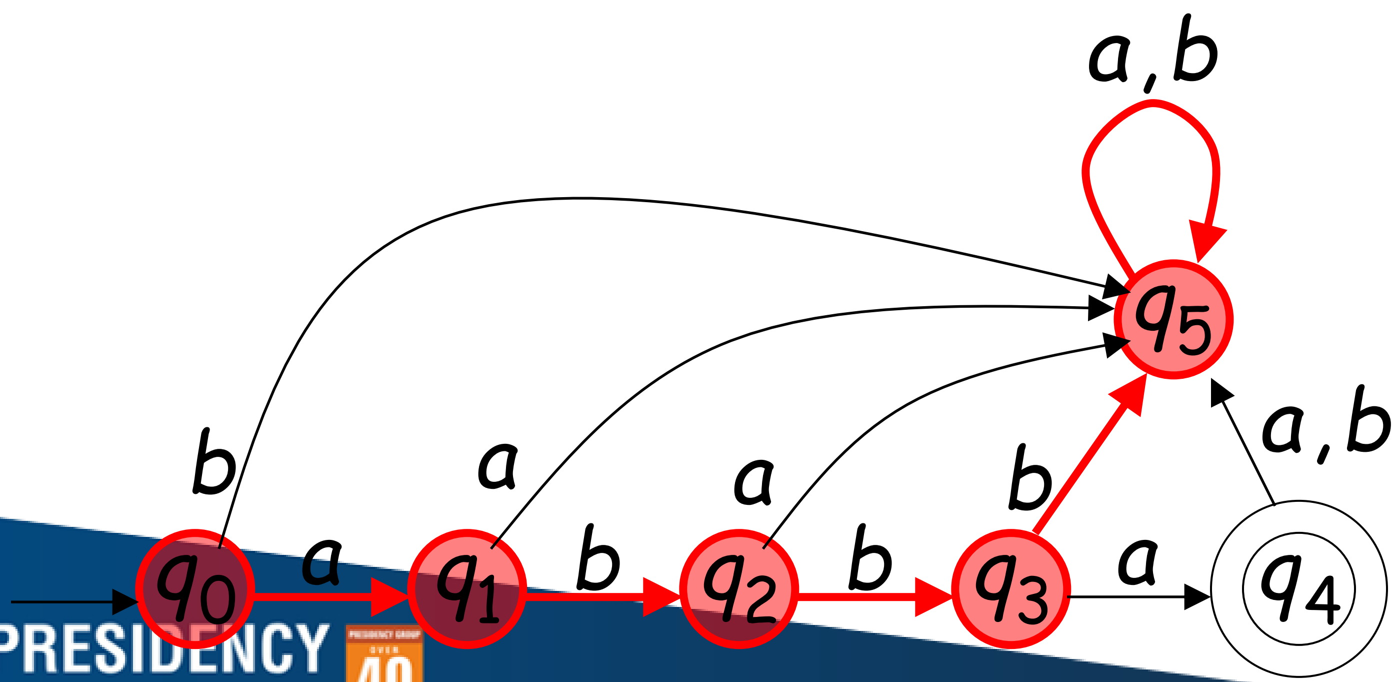
$$\delta^*(q_0, ab) = q_2$$



$$\delta^*(q_0, abba) = q_4$$

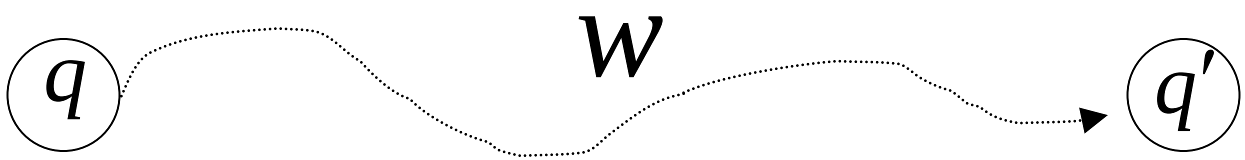


$$\delta^*(q_0, abbbaa) = q_5$$

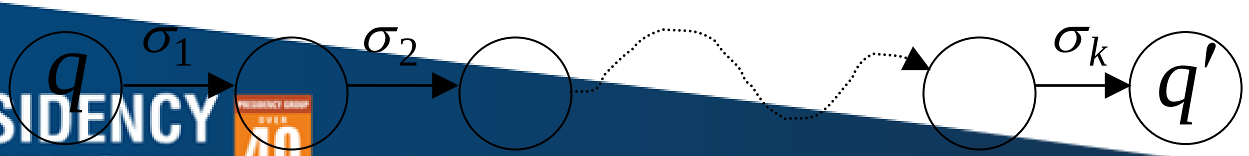


Observation: if there is a walk from q to q' with label w then

$$\delta^*(q,w) = q'$$



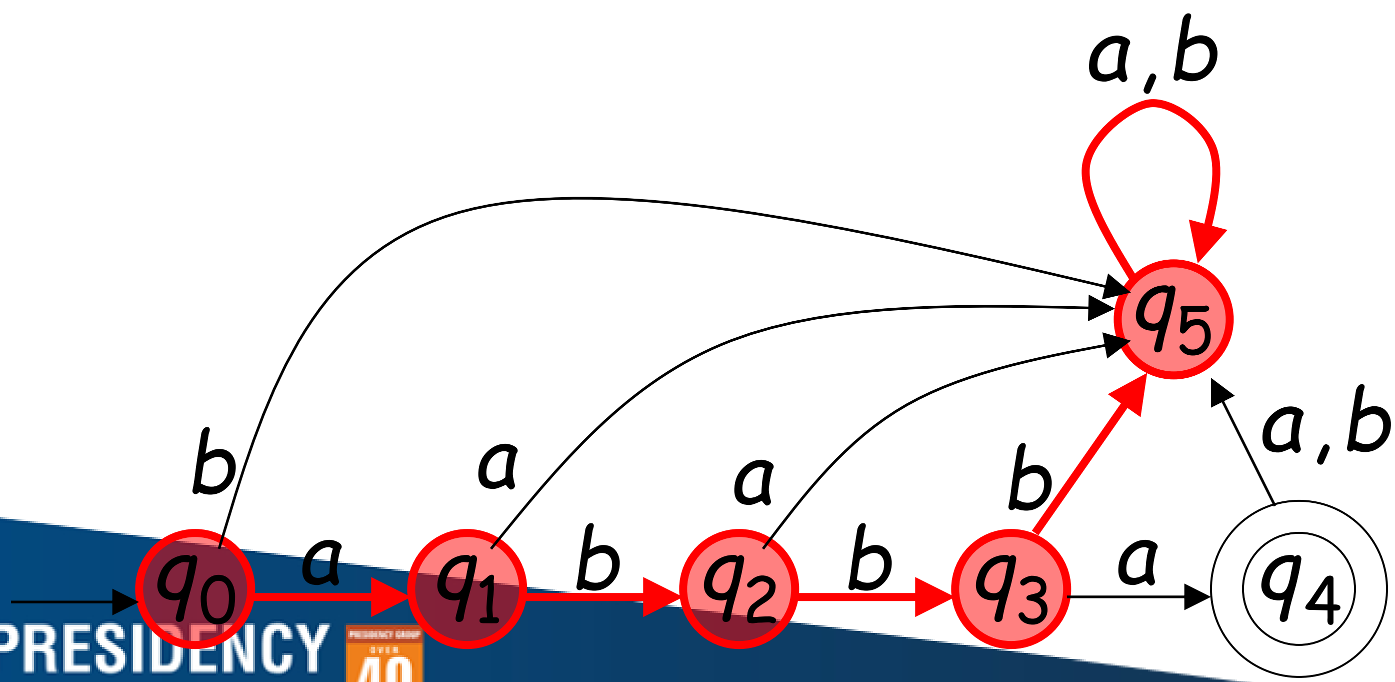
$$w = \sigma_1 \sigma_2 \cdots \sigma_k$$



PRESIDENCY UNIVERSITY

Example: There is a walk from q_0 to q_5 with label $abbbaa$

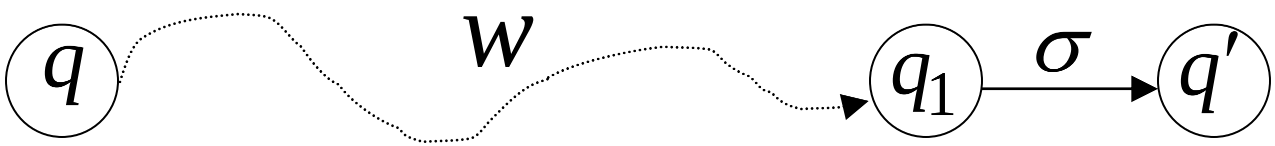
$$\delta^*(q_0, abbbaa) = q_5$$



Recursive Definition

$$\delta^*(q,\lambda) = q$$

$$\delta^*(q,w\sigma) = \delta(\delta^*(q,w),\sigma)$$



$$\left.\begin{array}{l}\delta^*(q,w\sigma) = q' \\ \\ \delta(q_1,\sigma) = q'\end{array}\right\} \Rightarrow \delta^*(q,w\sigma) = \delta(q_1,\sigma)$$

$$\left.\begin{array}{l}\delta^*(q,w\sigma) = \delta(q_1,\sigma) \\ \\ \delta^*(q,w) = q_1\end{array}\right\} \Rightarrow \delta^*(q,w\sigma) = \delta(\delta^*(q,w),\sigma)$$

PRESIDENCY UNIVERSITY

$$\delta^*(q_0, ab) =$$

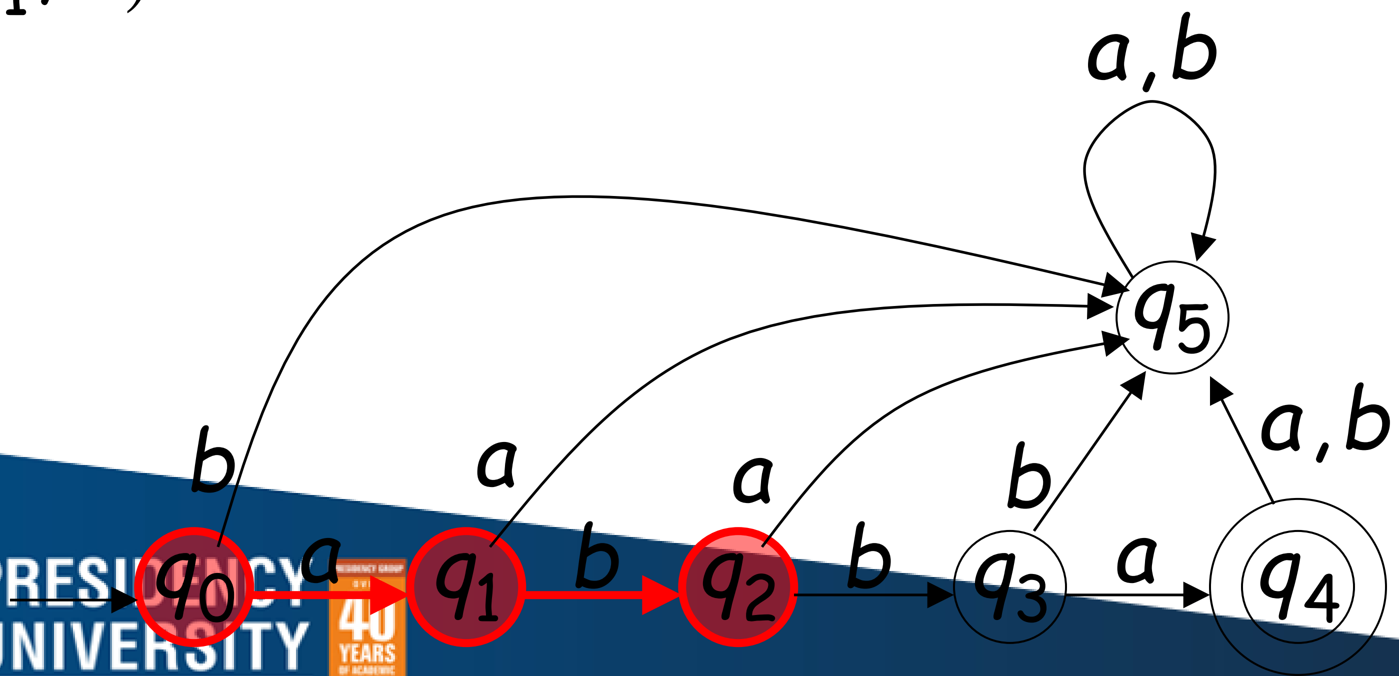
$$\delta(\delta^*(q_0, a), b) =$$

$$\delta(\delta(\delta^*(q_0, \lambda), a), b) =$$

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

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

$$q_2$$



Check the string acceptance of abba

- $\delta^*(qo,abba)= \delta(\delta^*(q0,abb) , a)$
- $= \delta(\delta (\delta^*(q0,ab),b) , a)$
- $= \delta(\delta (\delta (\delta^*(q0,a),b),b) , a)$
- $= \delta(\delta (\delta (\delta (\delta^*(q0,\lambda),a),b),b) , a)$
- $= \delta(\delta (\delta (\delta (q0,a),b),b) , a)$
- $=\delta(\delta (\delta (q1,b),b) , a)$
- $=\delta(\delta (q2,b) , a)$
- $=\delta(q3, a)$
- $=q4 \in F$
- String abba is accepted as q4 is a final state

PRESIDENCY UNIVERSITY

GAIN MORE KNOWLEDGE REACH GREATER HEIGHTS

Private University Estd. in Karnataka State by Act No. 41 of 2013

PRESIDENCY GROUP OVER 40 YEARS OF ACADEMIC WISDOM

Language Accepted by FAs

- For a FA $M=(Q,\Sigma,\delta,q_0,F)$

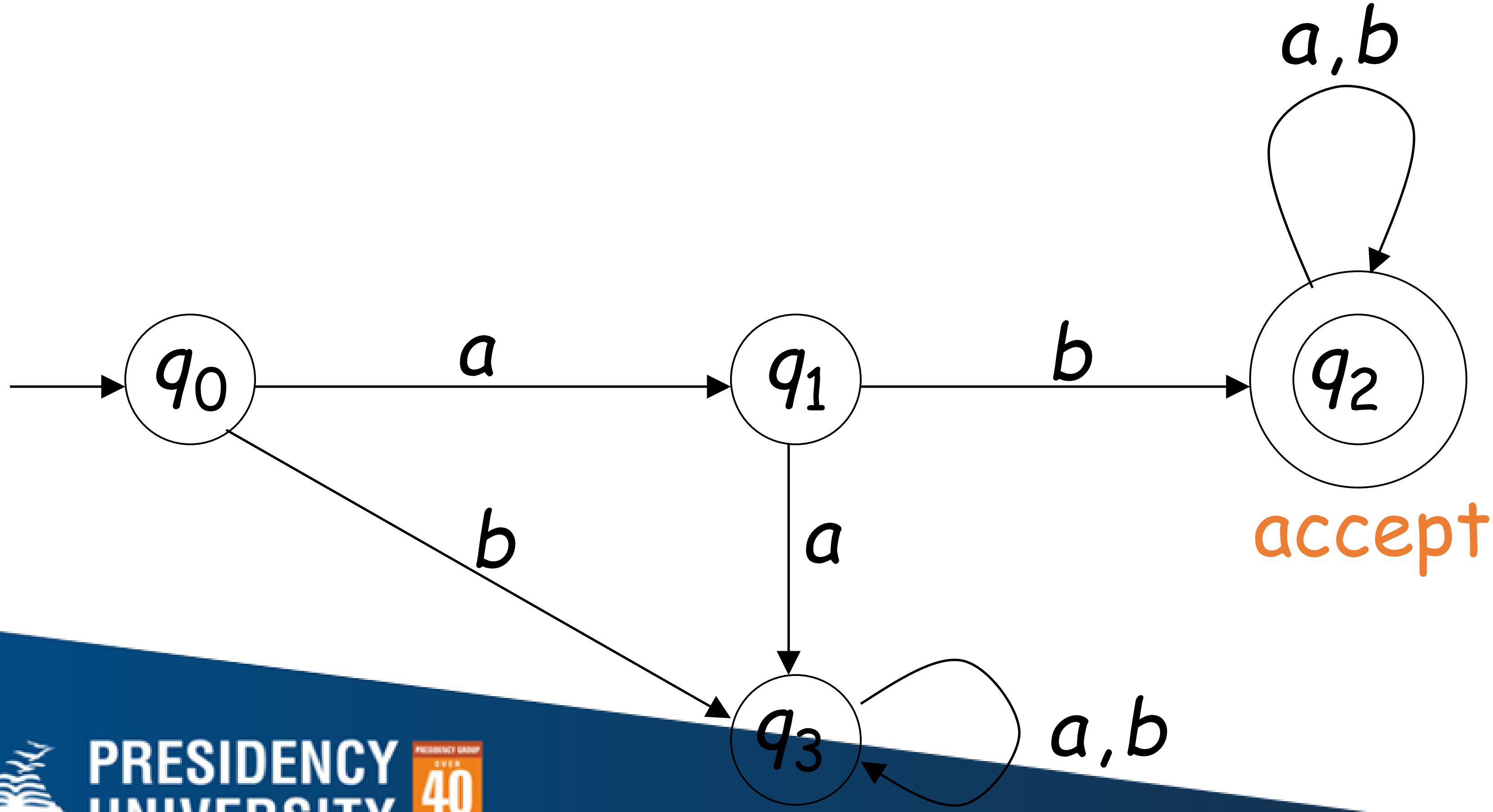
- Language accepted by M :

- $$L(M)=\{w\in\Sigma^*:\delta^*(q_0,w)\in F\}$$

$q'\in F$

Example

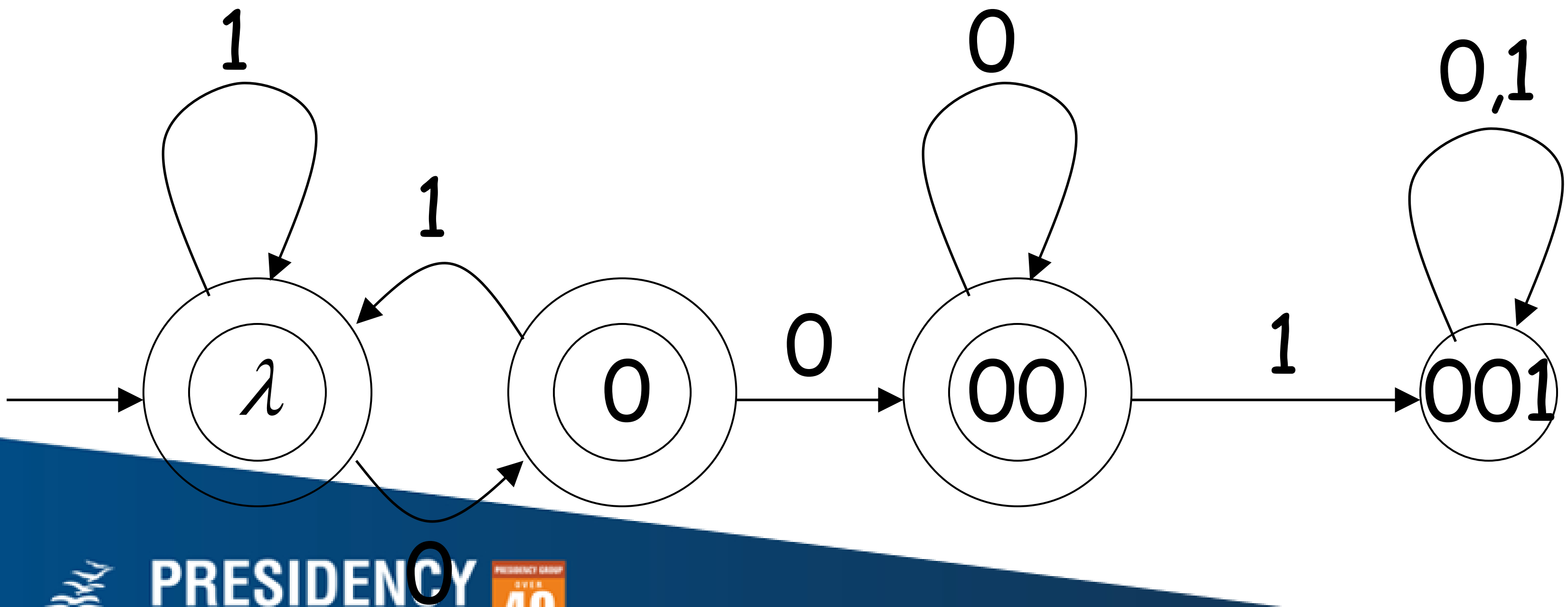
- $L(M)$ = { all strings with prefix ab }



L=strings with substring ‘101’ over {0, 1}

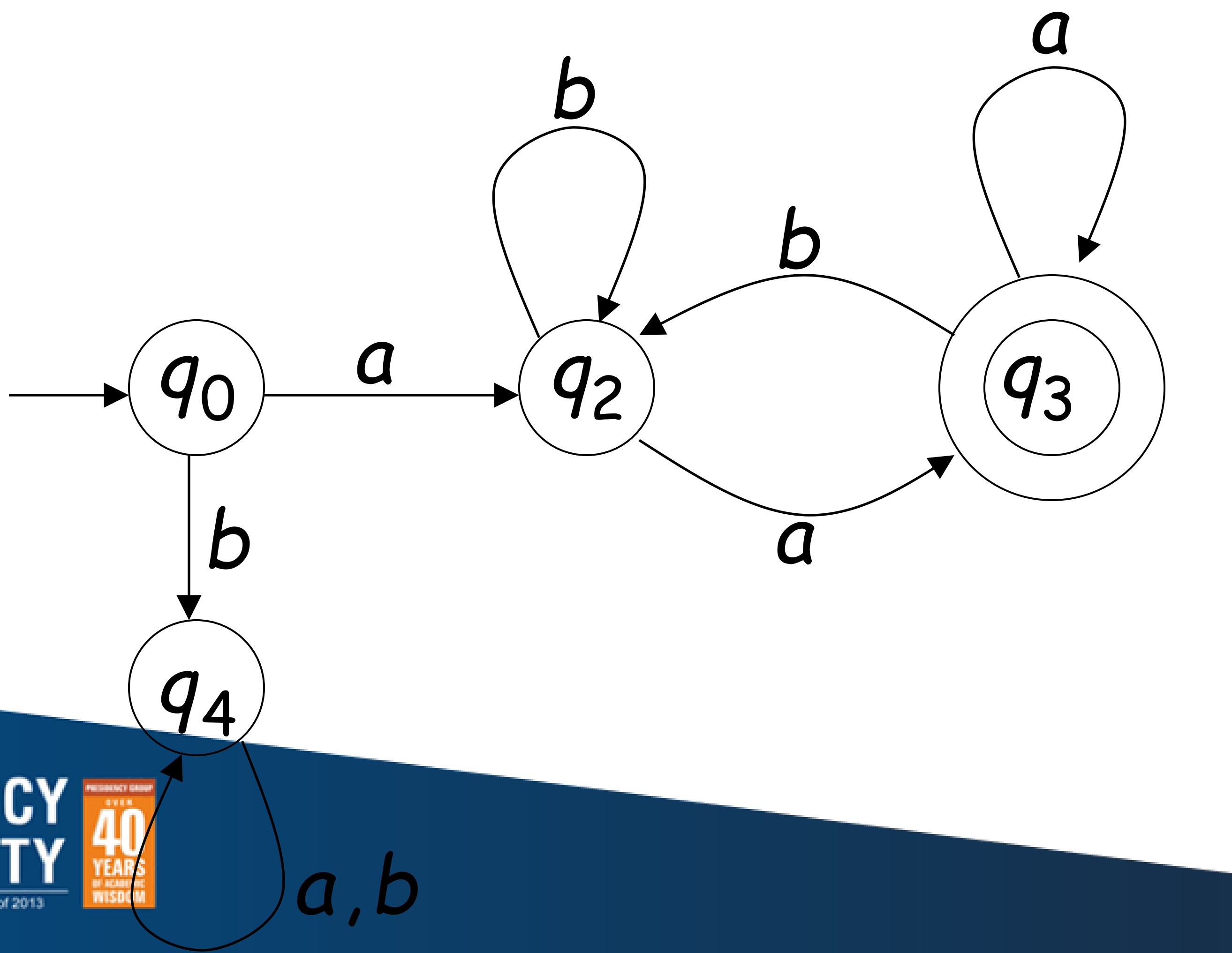
Example

- $L(M) = \{$ all strings without substring 001 $\}$



Example

$$L(M) = \{awa : w \in \{a,b\}^*\}$$

- 

Regular Languages

- Definition:
- A language L is regular if there is
- FA M such that $L = L(M)$

- Observation:
- All languages accepted by FAs
- form the family of regular languages
  -

Examples of regular languages:

$\{abba\}$ $\{\lambda, ab, abba\}$

$\{awa : w \in \{a,b\}^*\}$ $\{a^n b : n \geq 0\}$

{ all strings with prefix ab }

{ all strings without substring 001 }

There exist automata that accept these Languages

PRESIDENCY UNIVERSITY

GAIN MORE KNOWLEDGE REACH GREATER HEIGHTS

Private University Estd. in Karnataka State by Act No. 41 of 2013

There exist languages which are <u>not</u> Regular:

Example: $L = \{a^n b^n : n \geq 0\}$

There is no FA that accepts such a language

(we will prove this later in the class)

Deterministic Finite Automata

- Every State should have transition over every input symbol
- There should be only One next state for each transition
- DFA is defined as

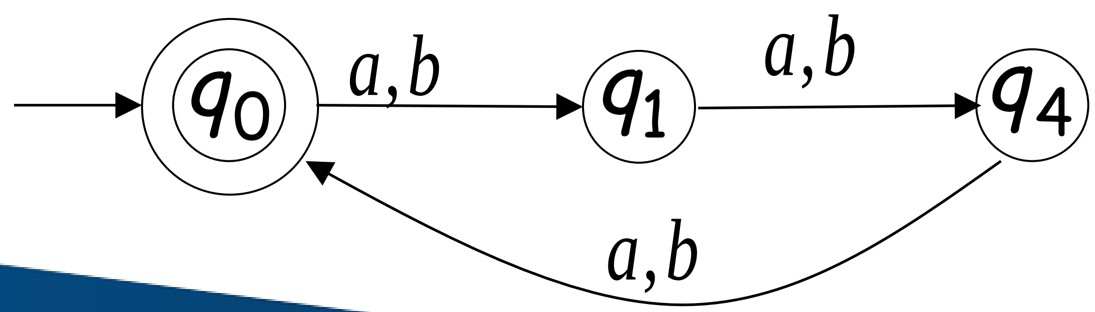
$$M = (Q, \Sigma, \delta, q_0, F)$$

DFA Example

Design DFA that accepts language L= { w : |w| mod 3 = 0 } over ∑ = {a, b}

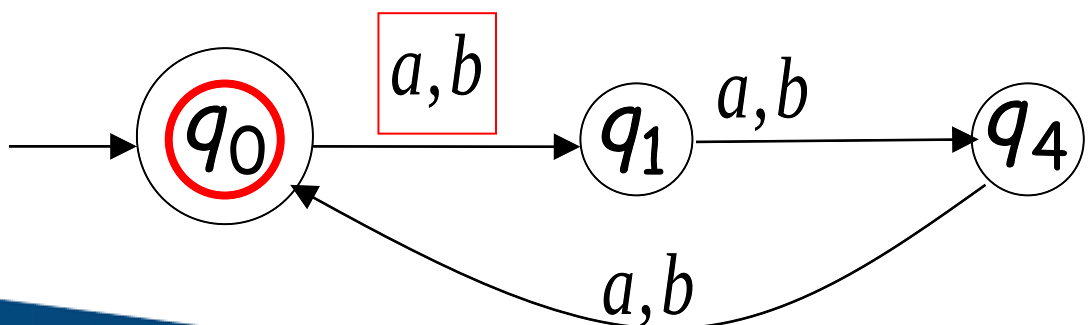
Solution

- **Strings accepted= {ϵ, aaa, bbb, aba, aab, bab, aaabbb, ababab, ...}**
- **Strings rejected= {a, b, ab, ba, abab, baba, bbaa, aaabb, …}**
- **Transition Diagram**



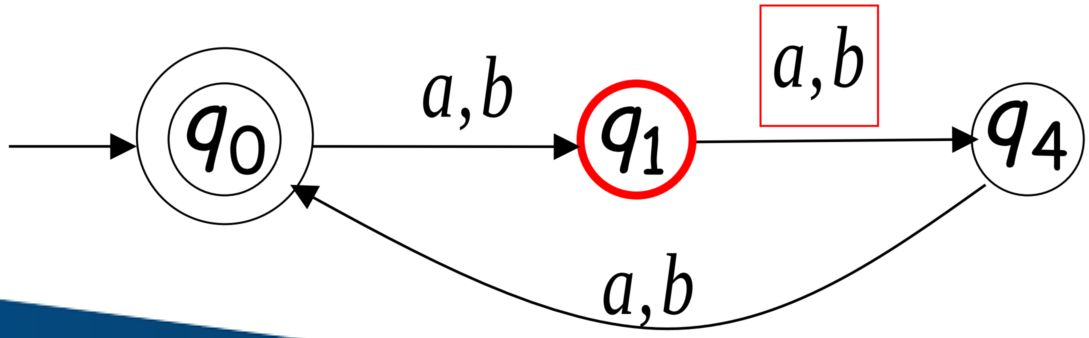
DFA Example

- Consider Sample String : **a a b**



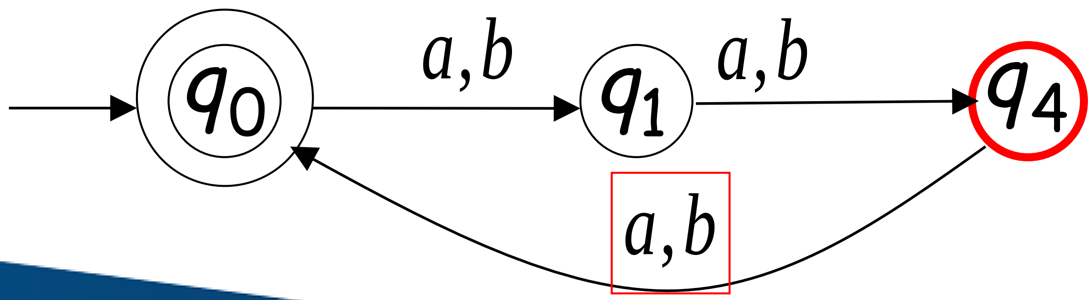
DFA Example

- Consider Sample String : **a a b**



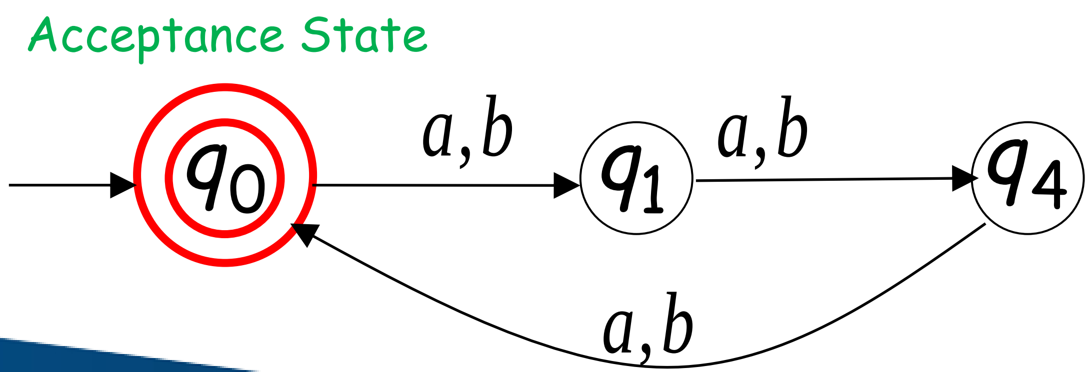
DFA Example

- Consider Sample String : **a a b**



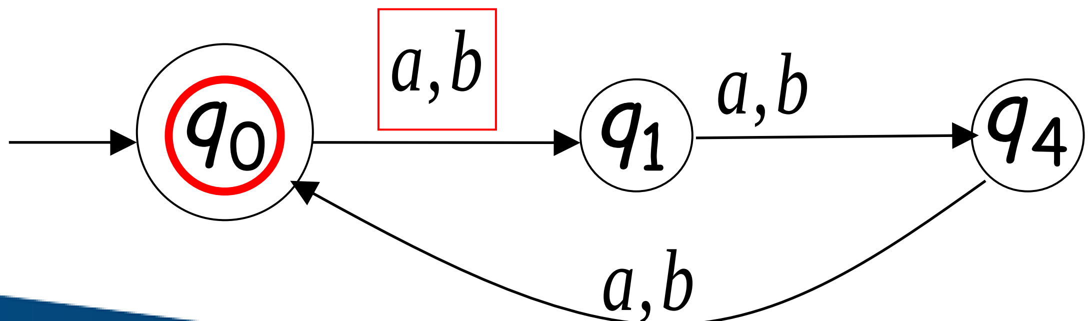
DFA Example

- ➢Consider Sample String : **a a b**
- ➢ String is accepted



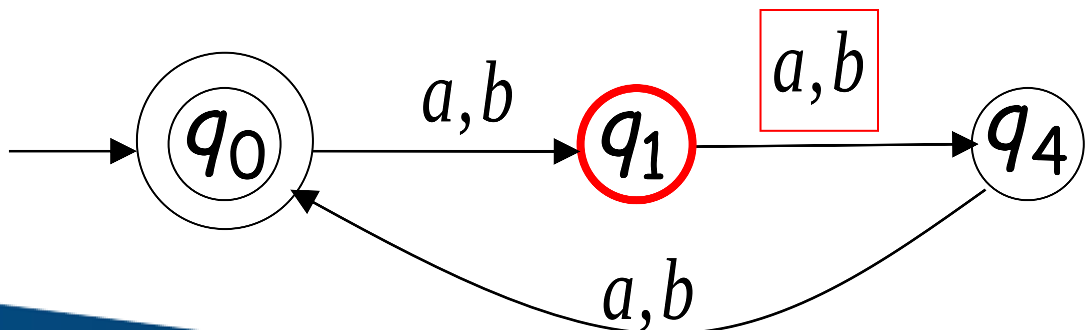
DFA Example

- Consider Another Sample String : **b a b a**



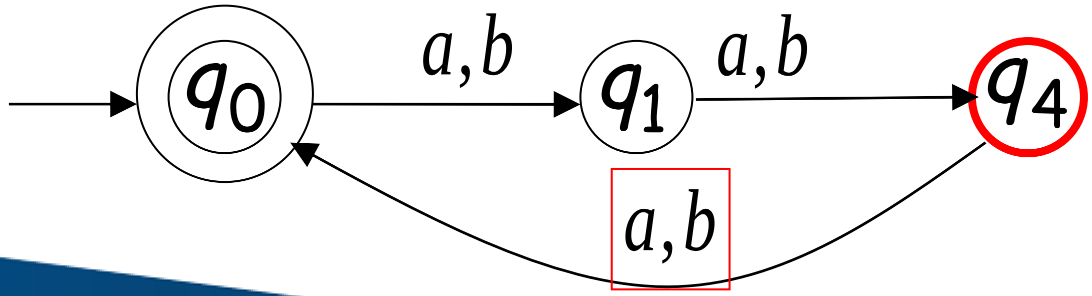
DFA Example

- Consider Another Sample String : **b a b a**



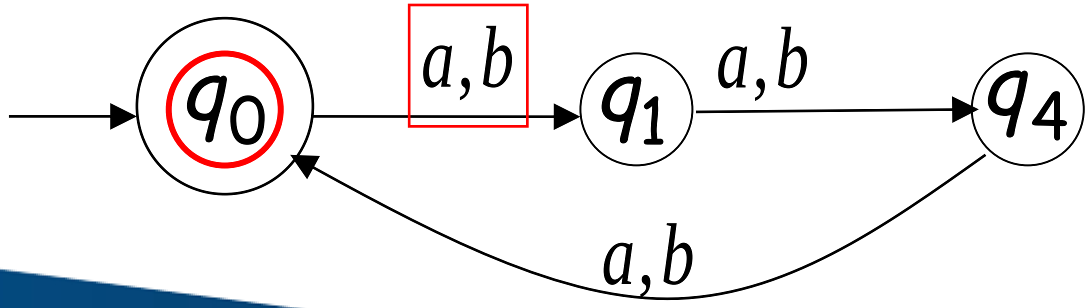
DFA Example

- Consider Another Sample String : **b a b a**



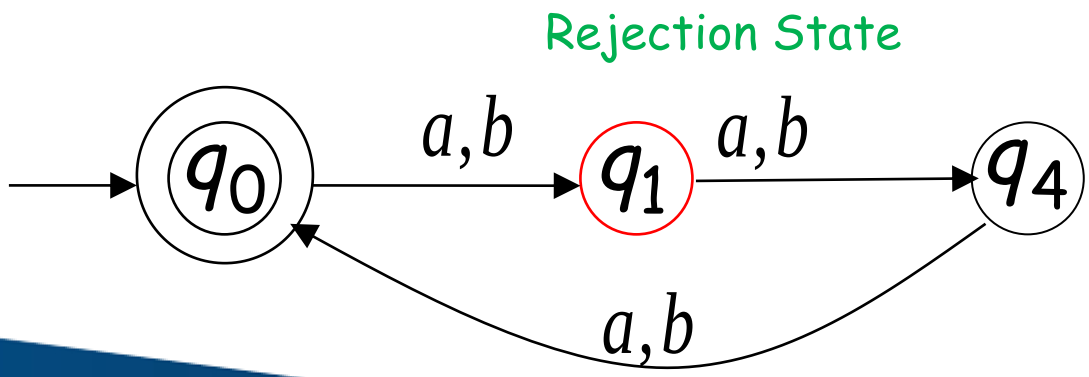
DFA Example

- Consider Another Sample String : **b a b a**



DFA Example

- ➢ **Consider Another Sample String :** **b a b a**
- ➢ **String is rejected**



DFA Example

- DFA Tuples $M=(Q, \Sigma, \delta, q0, F)$
- $Q= \{q0, q1, q4\}$
- $\Sigma= \{a, b\}$
- δ = Transition function represented by Transition table
- q0= Initial State
- F= {q0} → Acceptance State

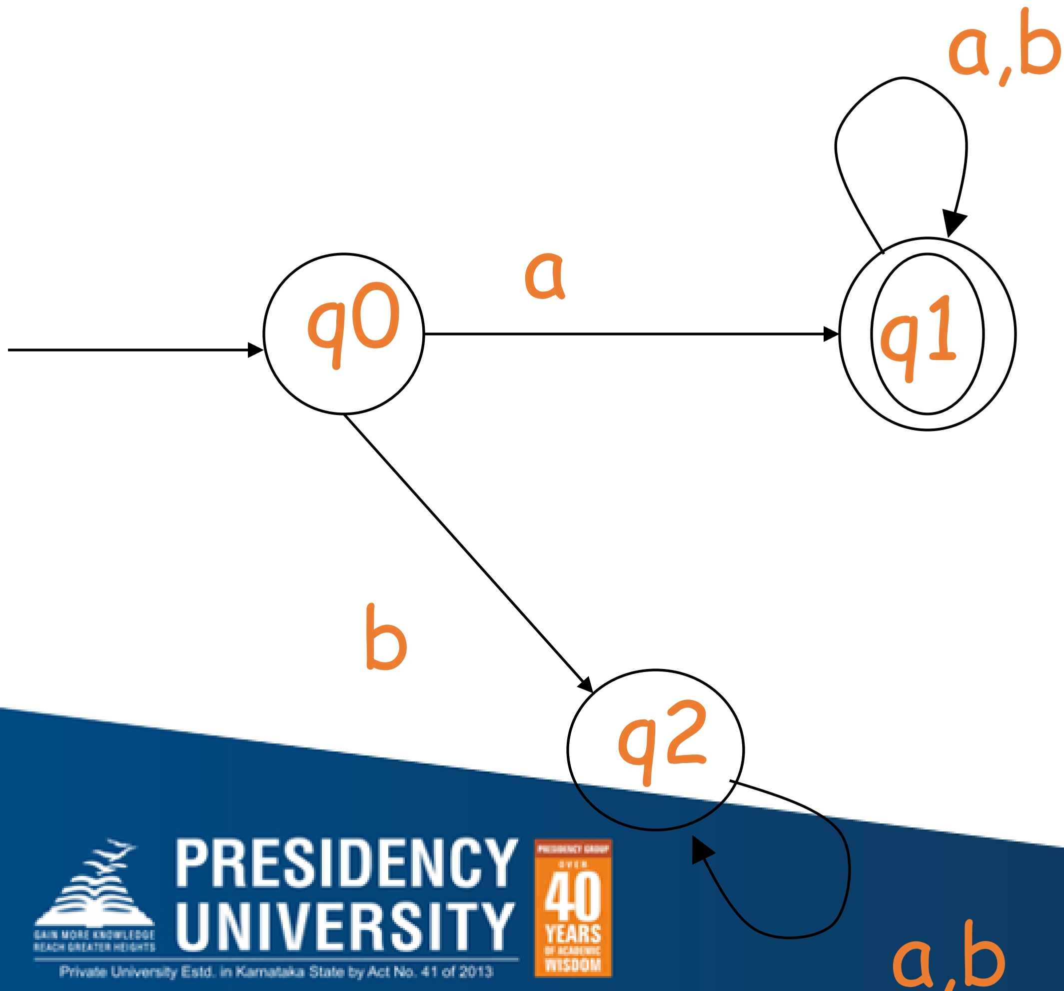
DFA Example

➢Transition Table

Σ Q	a	b
→*q0	q1	q1
q1	q4	q4
q4	q0*	q0*

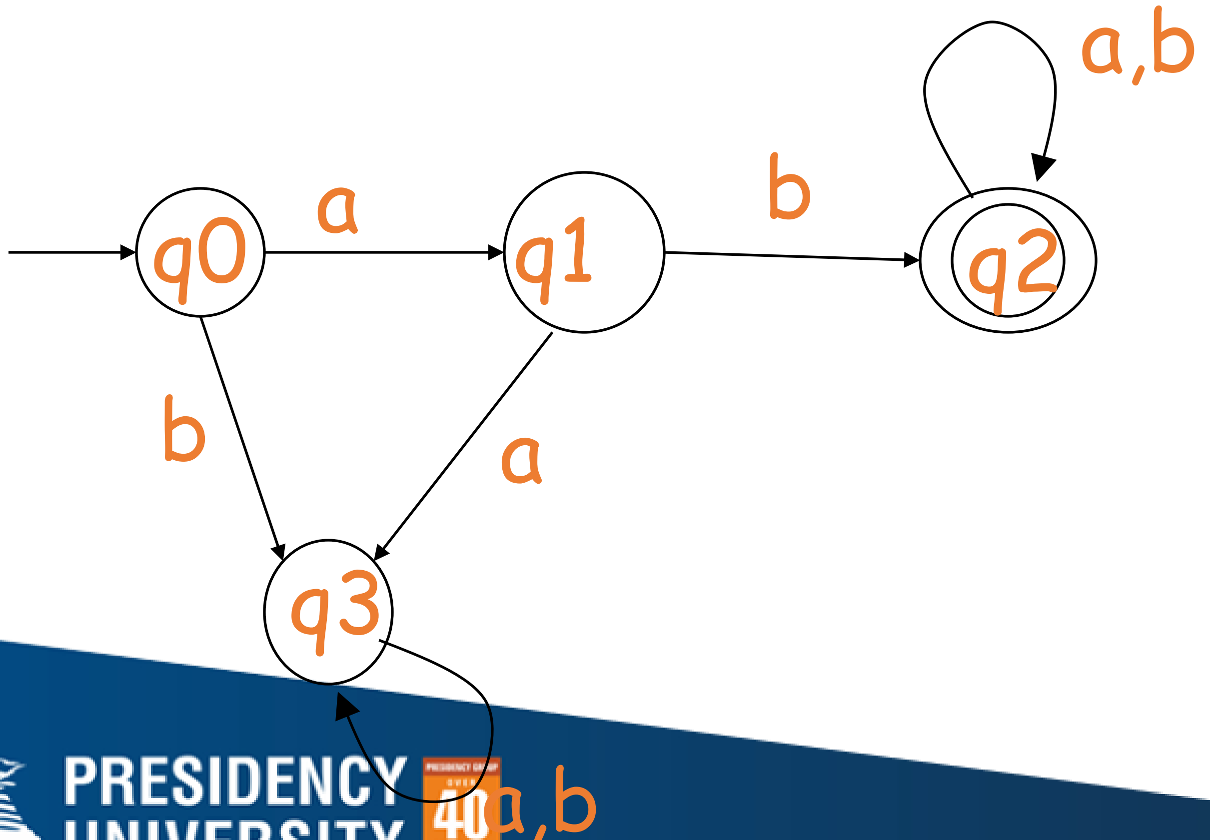
DFA - Starts with a , $\sum$={a,b}

- L= { a,aa,ab,aaa,aab,aba,….}



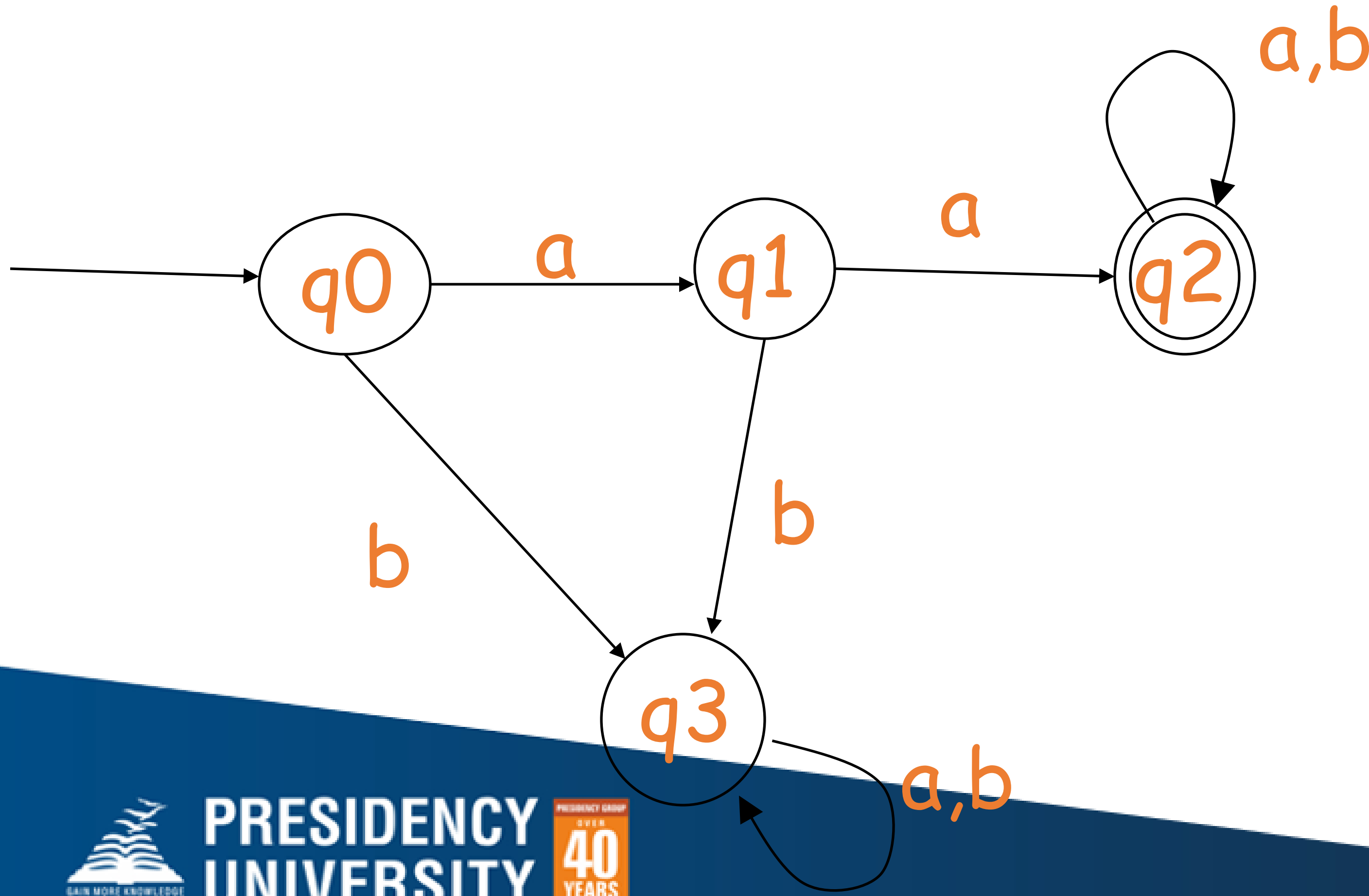
DFA - Starts with ab , ∑={a,b}

- L= { ab,aba,abb,….}



DFA - Starts with aa , $\Sigma=\{a,b\}$

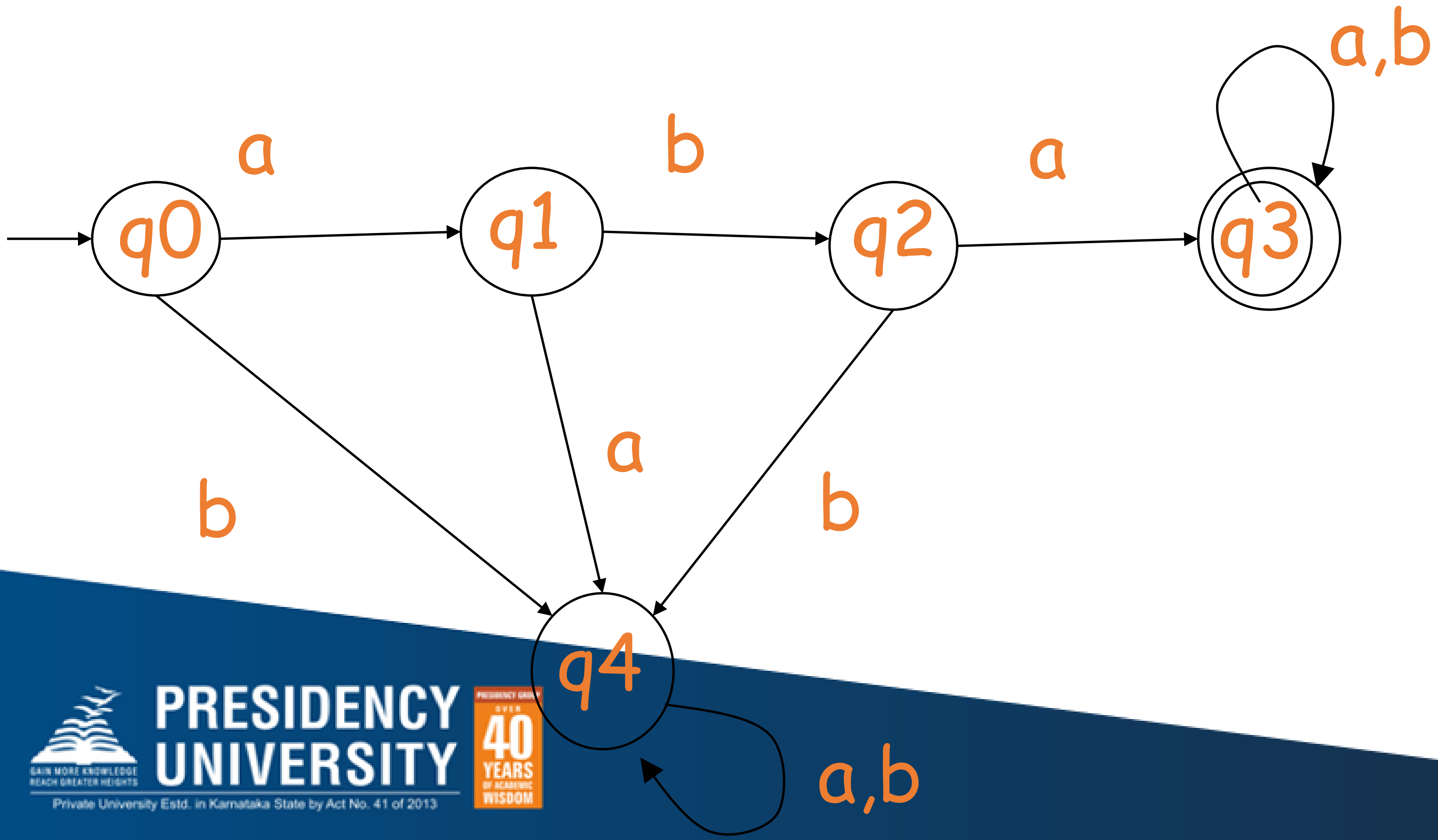
- L = { aa,aab,aaba,....}



PRESIDENCY UNIVERSITY

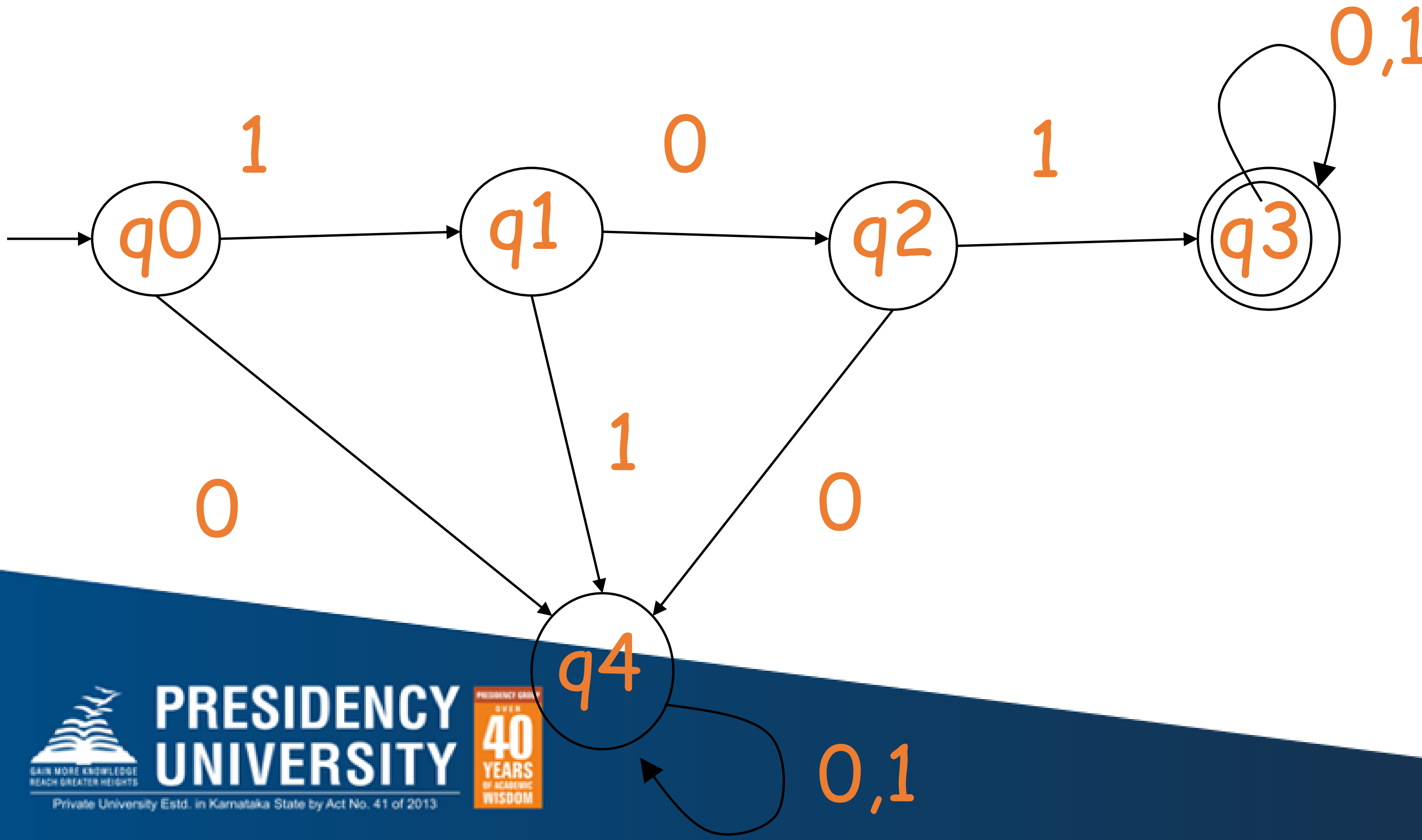
DFA - Starts with aba , $\sum$={a,b}

- L={ aba,abaa,abab,abaaa,…}



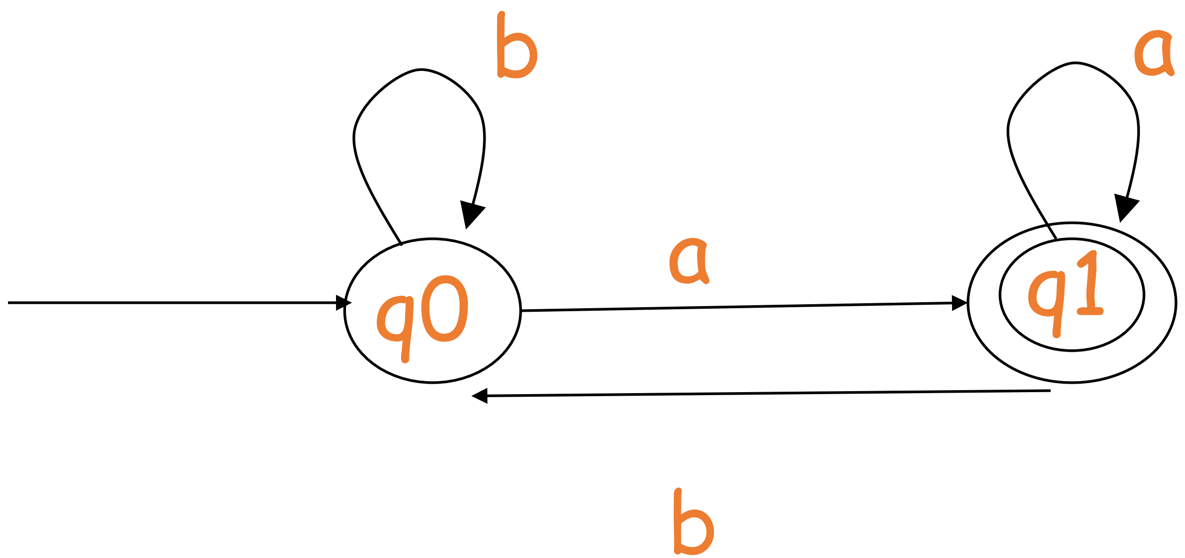
DFA - Starts with 101 , ∑={0,1}

- L={ 101,1010,1011,101101,...}



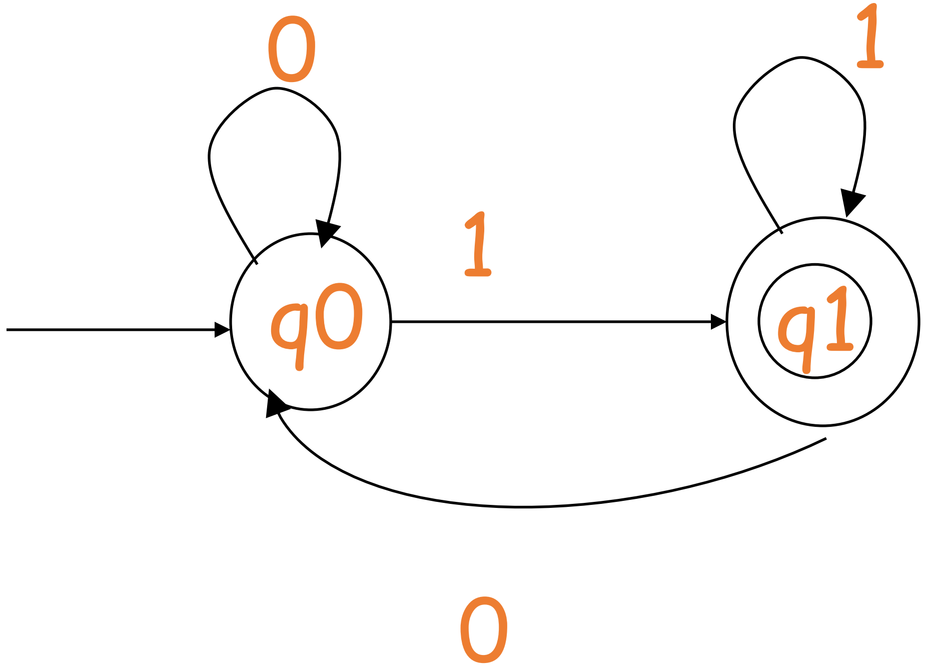
Ends with a

- L={a,aa,ba,aaa,aba,...}



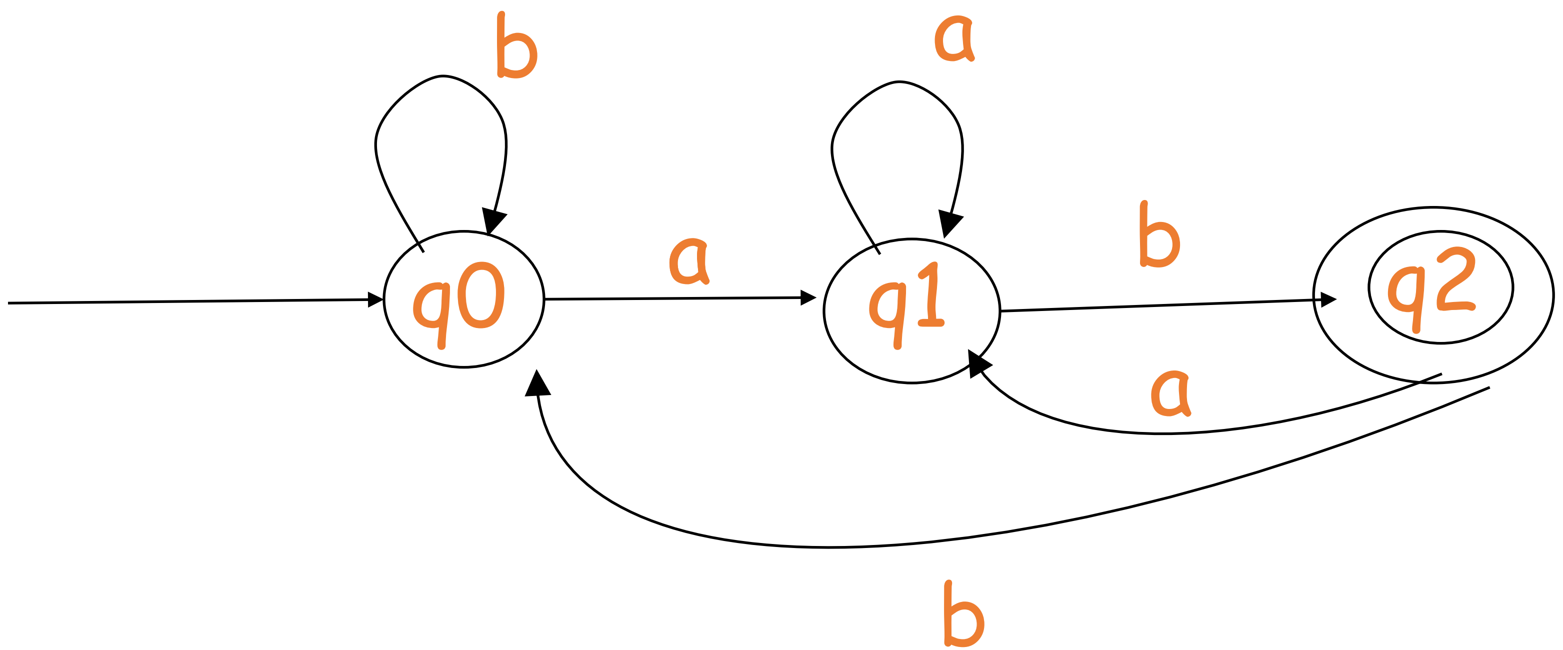
Ends with 1 , ∑={ 0,1)

- L={ 1,01,11,...}

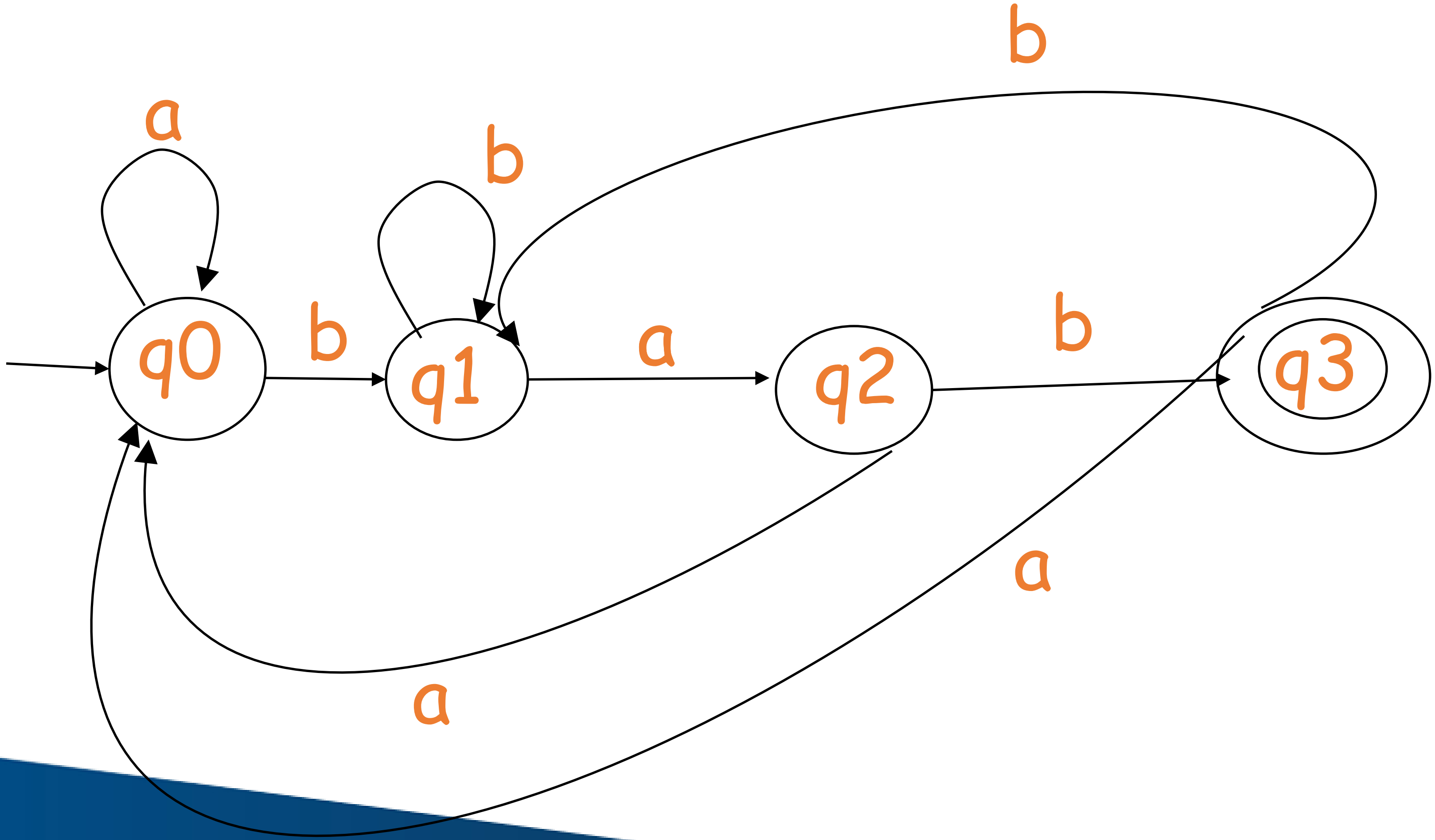


Ends with ab

- L={ab,aab,bab,aaab,bbbbab,abab,…}



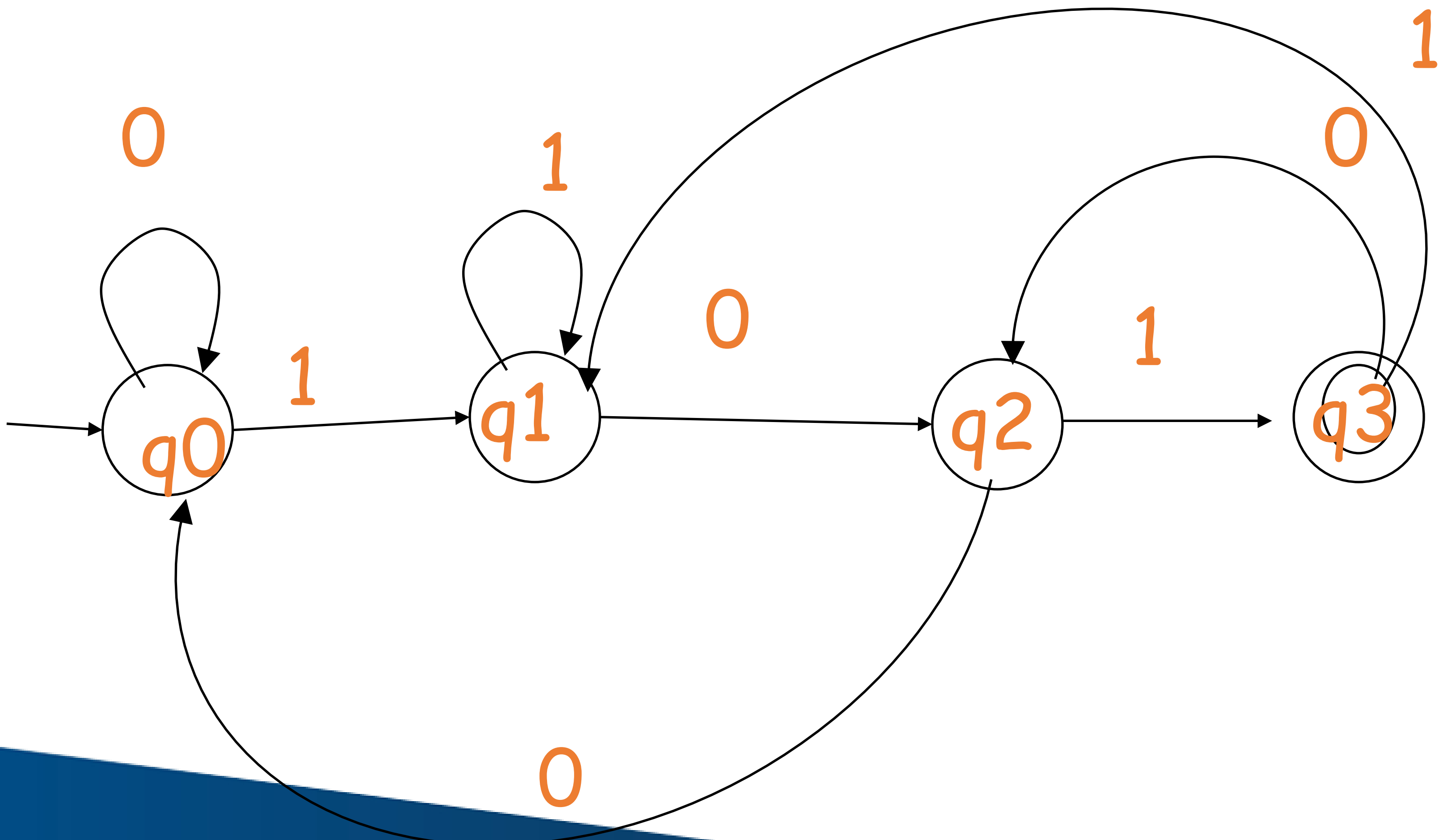
Ending with bab



120

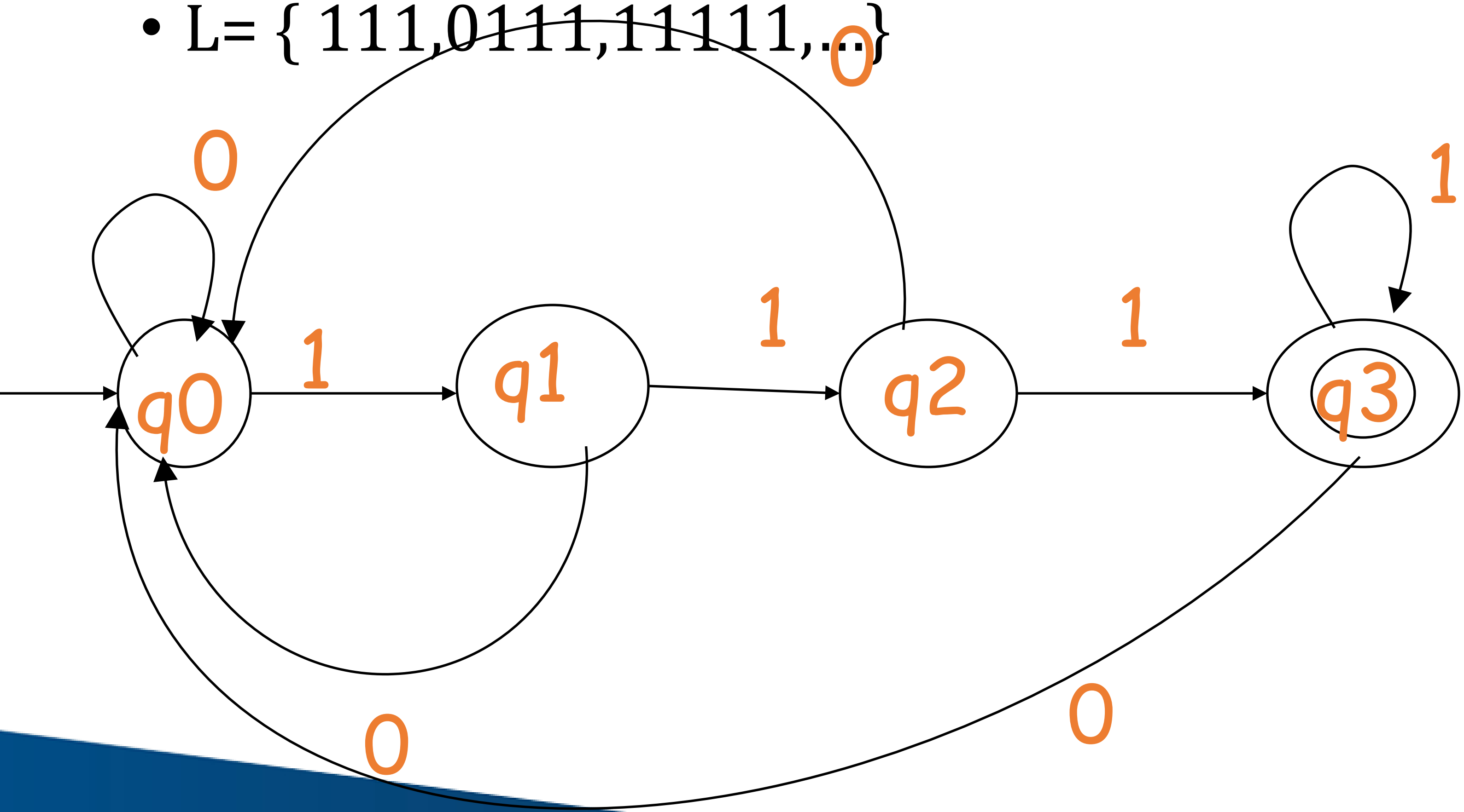
DFA – Ends with 101, Σ={ 0,1}

- L={ 101,0101,1101,000101,111101,…}



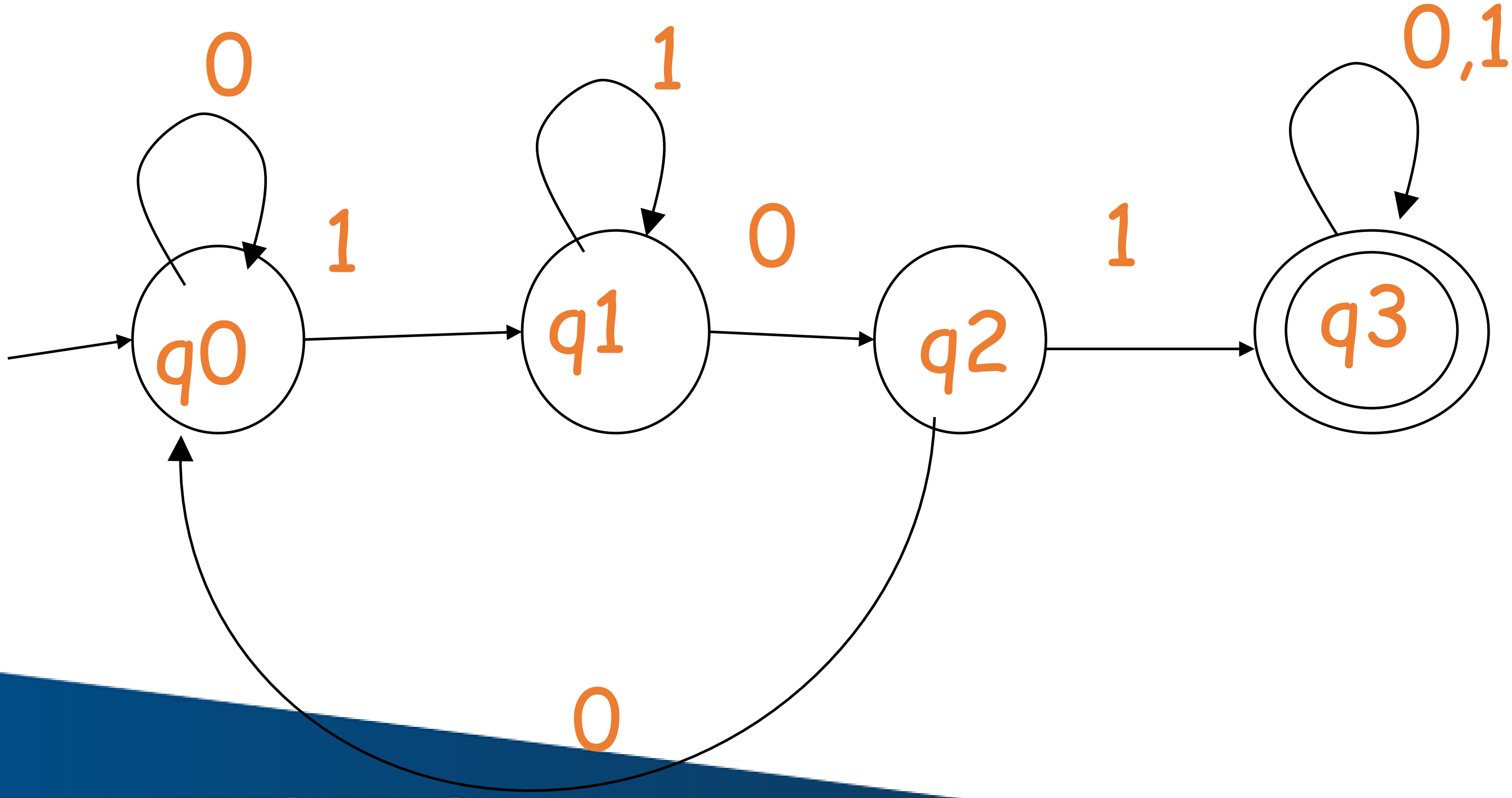
DFA – Ends with 111 , ∑={ 0,1}

- L= { 111,0111,11111,...}



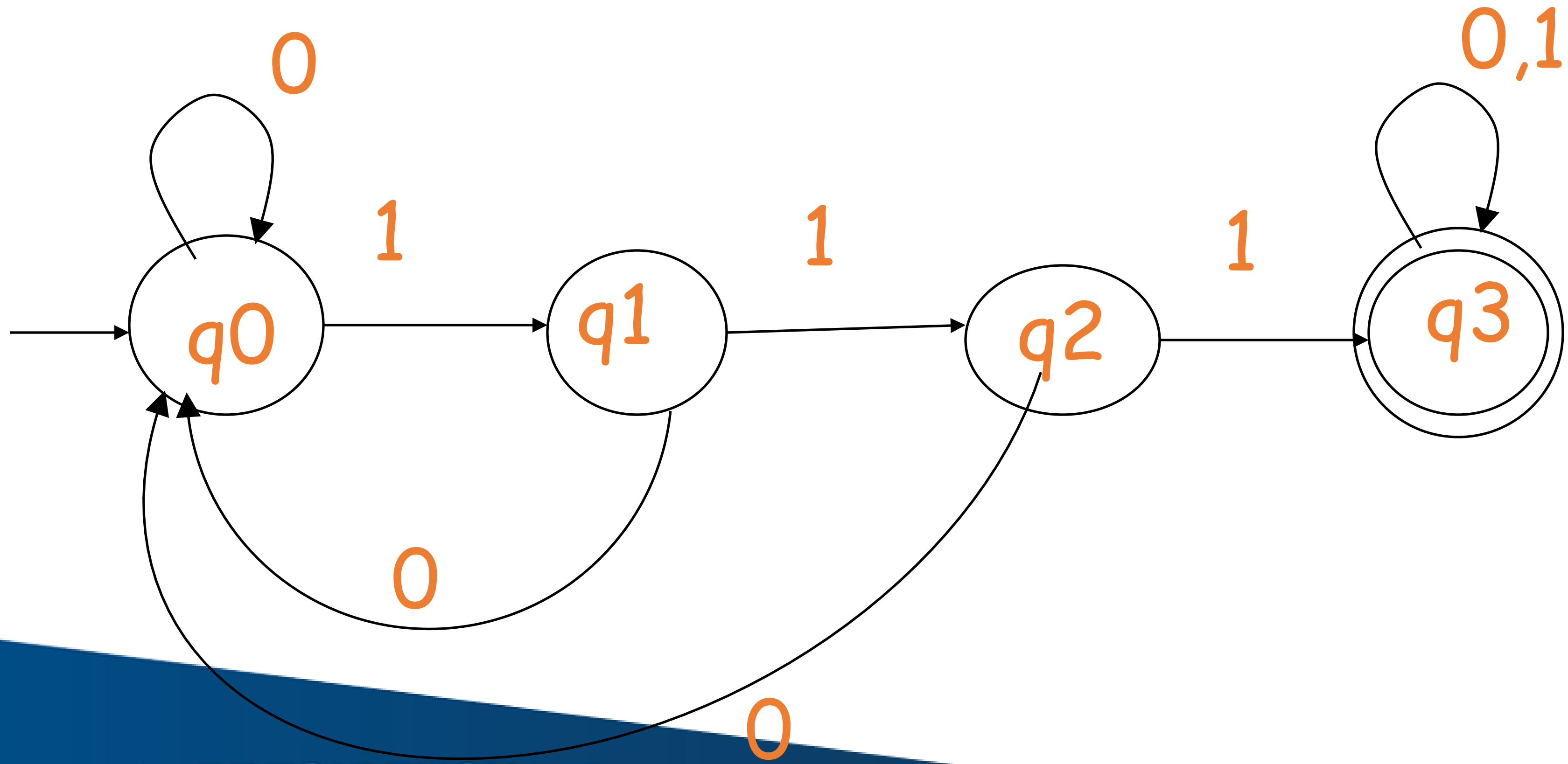
DfA – Substring 101

- L={ 101,0101,1101,1010,1011,00010111,...}



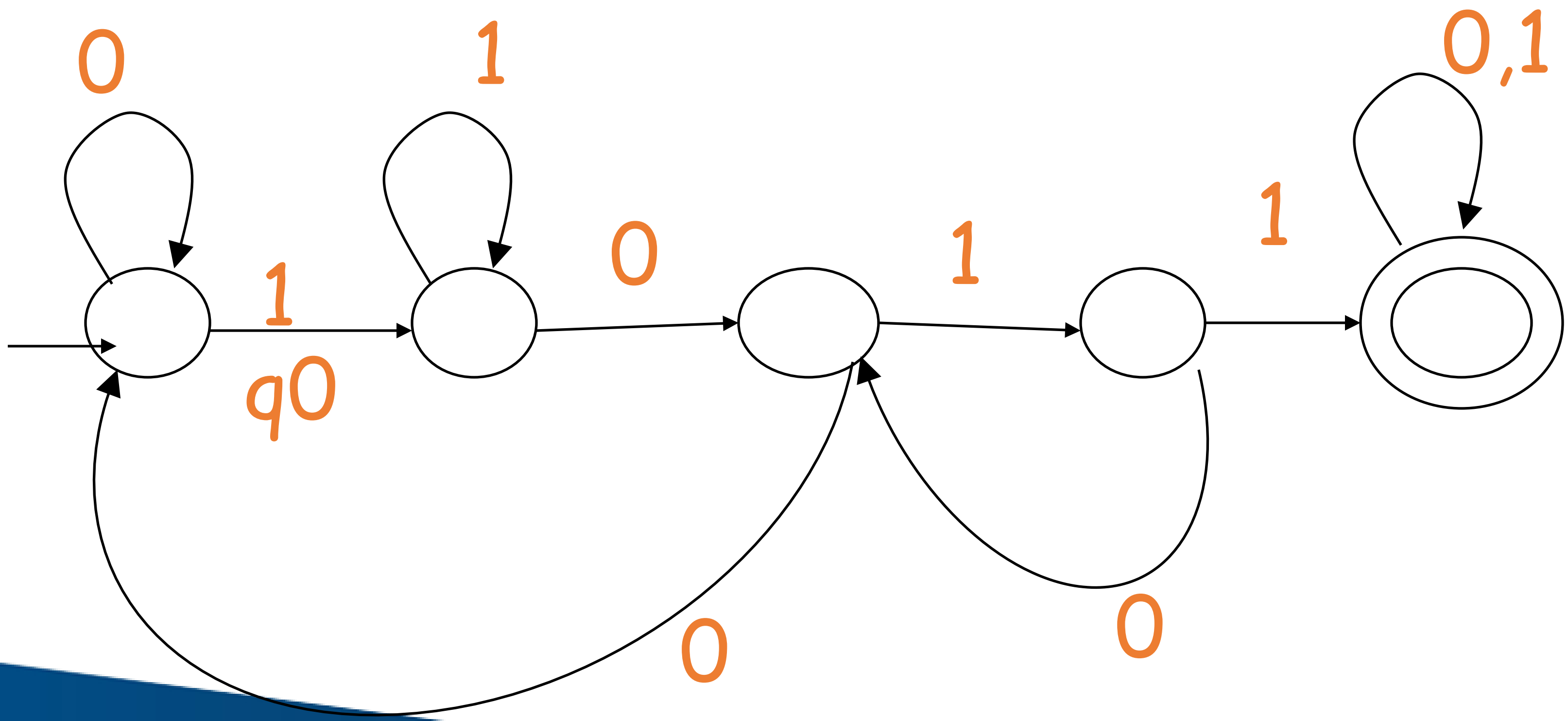
DFA- Contains 111

- L={ 111,0111,1111,000111000,11111000,..}



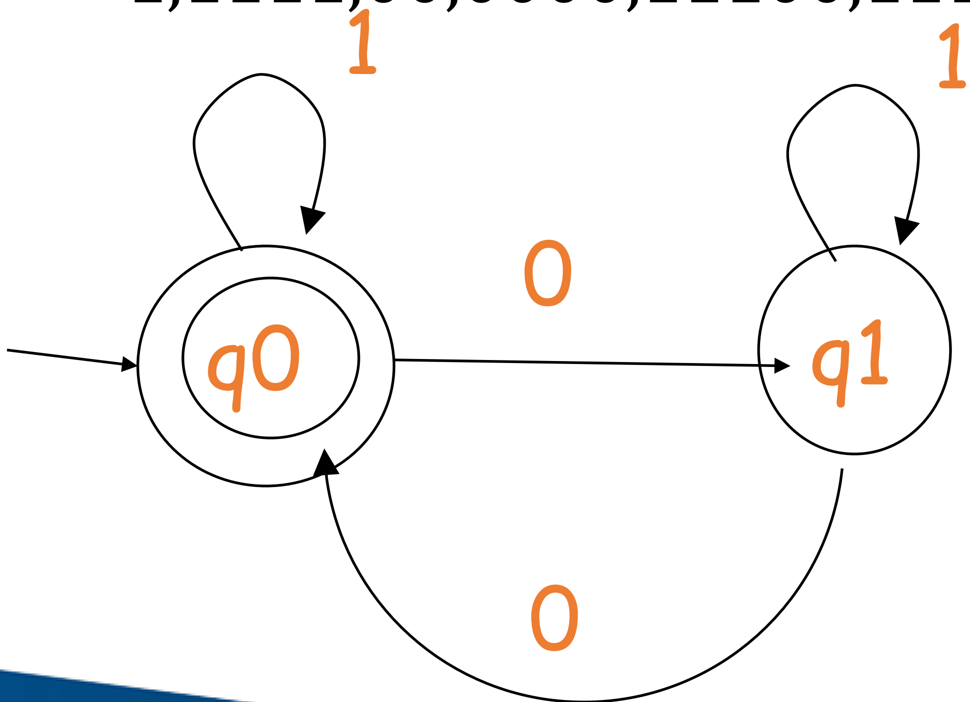
DFA – Contains 1011

- L= { 1011,01011,11011,00010111,..}



DFA- Even no of 0s, ∑={ 0,1} or n0(w)mod2=0

- L={ λ, 1,1111,00,0000,11100,1110000,0000111,..}

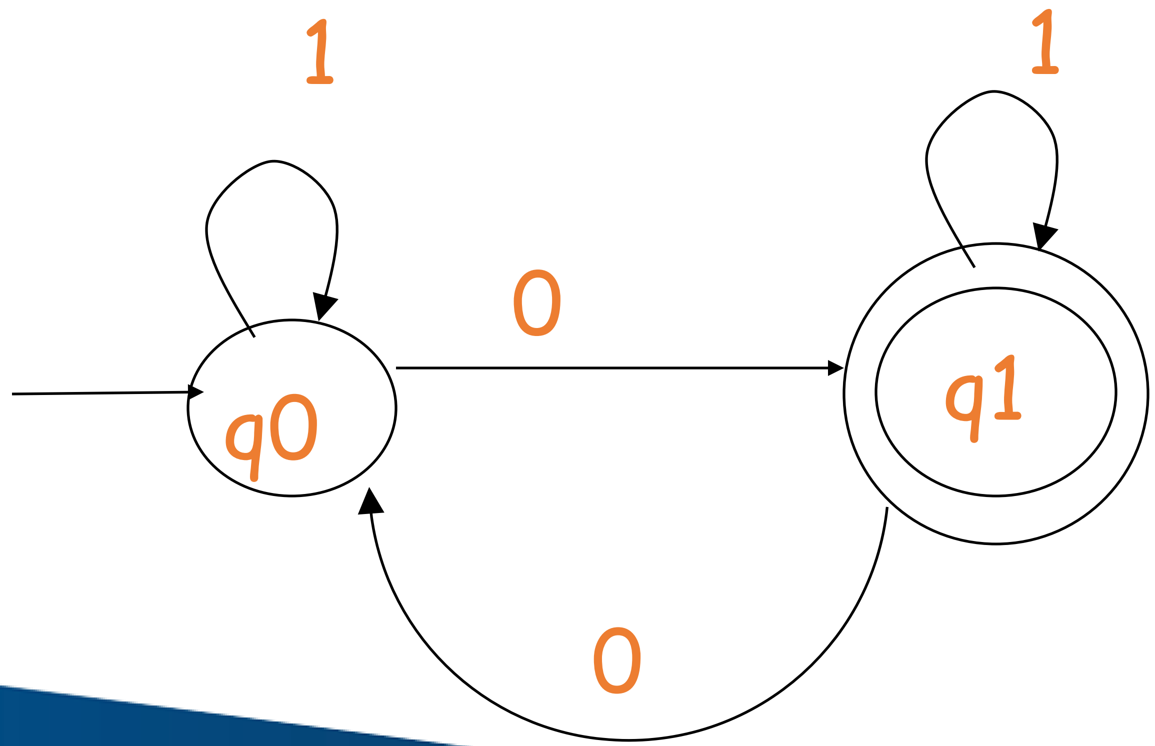


Transition	o	1
*->q0	q1	q0
q1	qo	q1

DFA- strings with even number of 0’s over {0, 1}

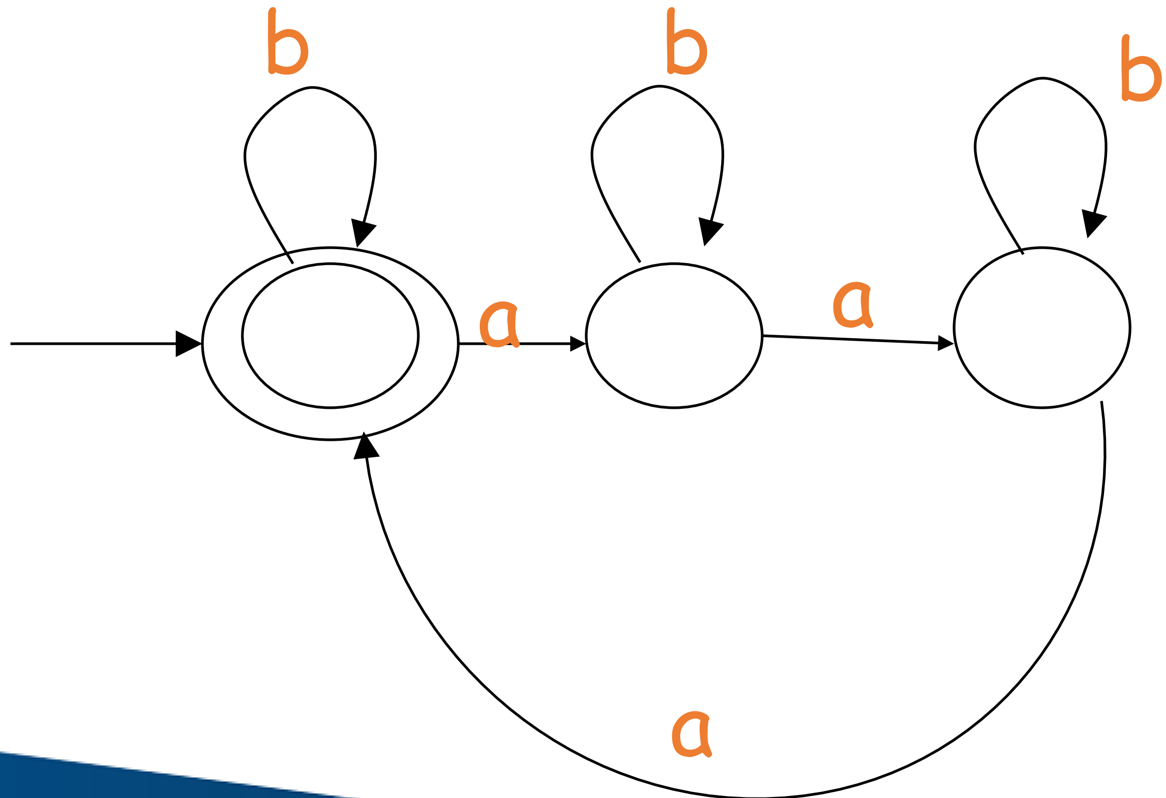
DFA – Odd no of 0's , ∑={ 0,1} or n0(w)mod2=1

- L={ 0,01,1110,000,000001111,...}

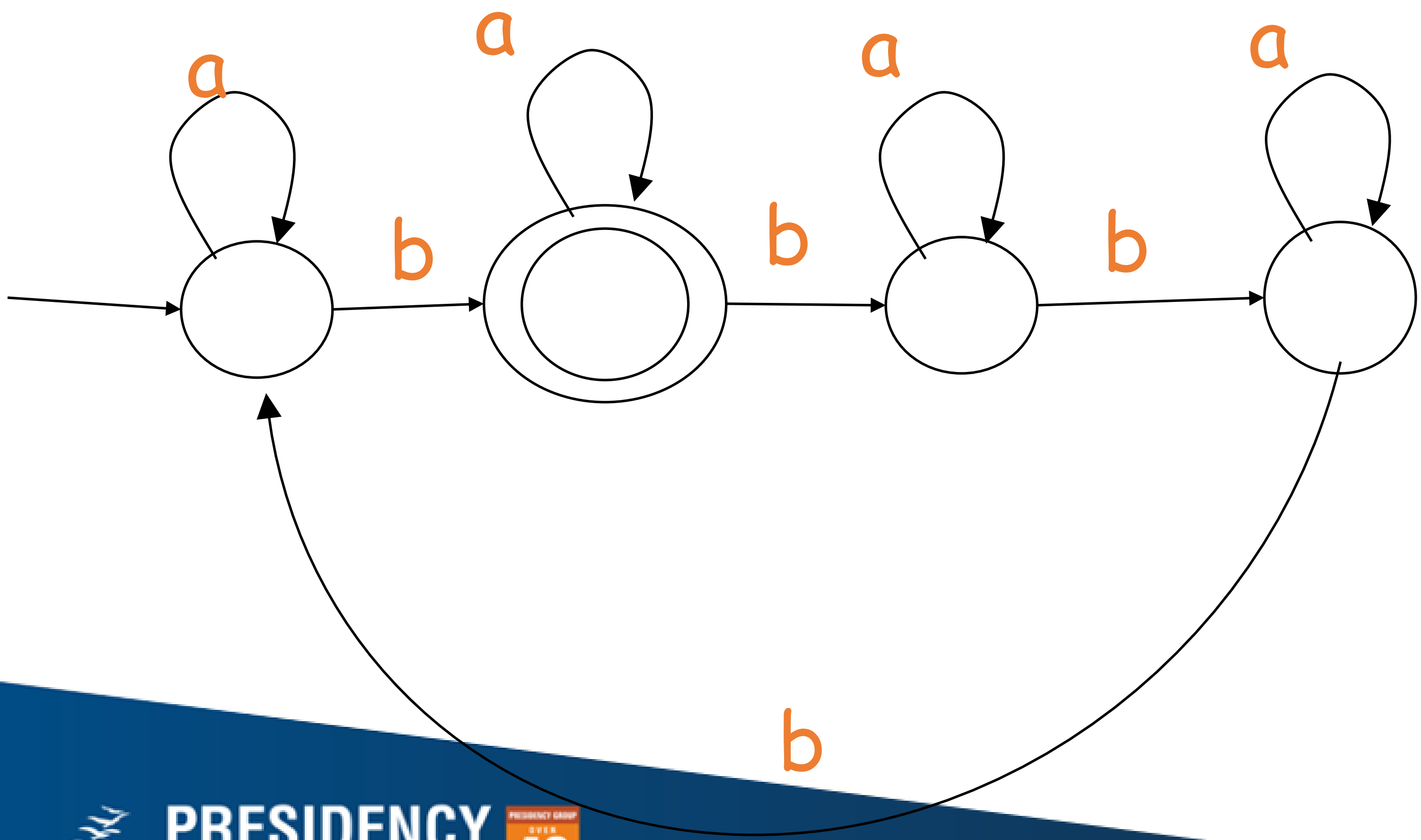


DFA - na(w)mod3=0,∑={ a,b}

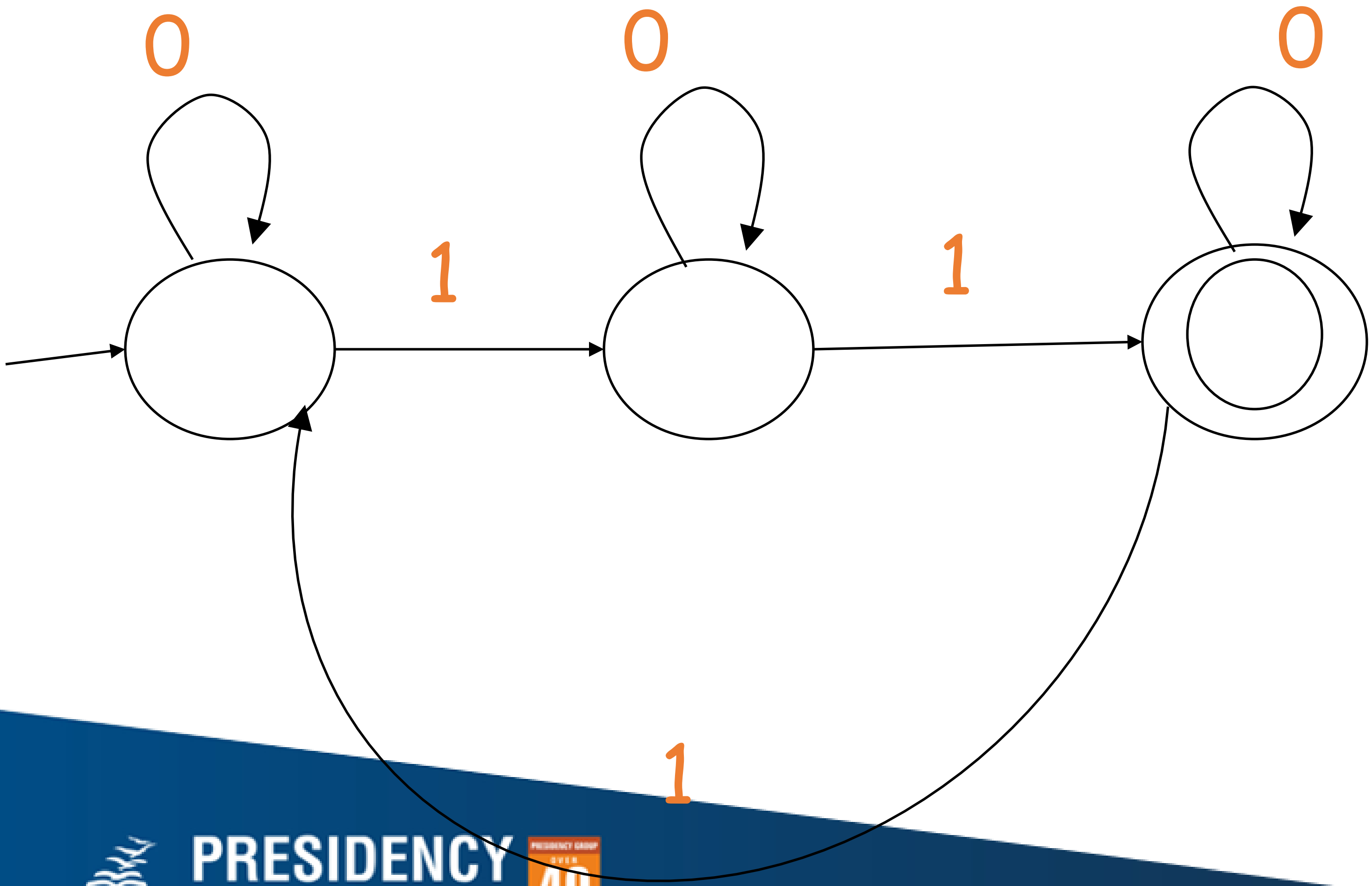
L={ λ,bbbb,aaabbb,aaaaaabbb,....}



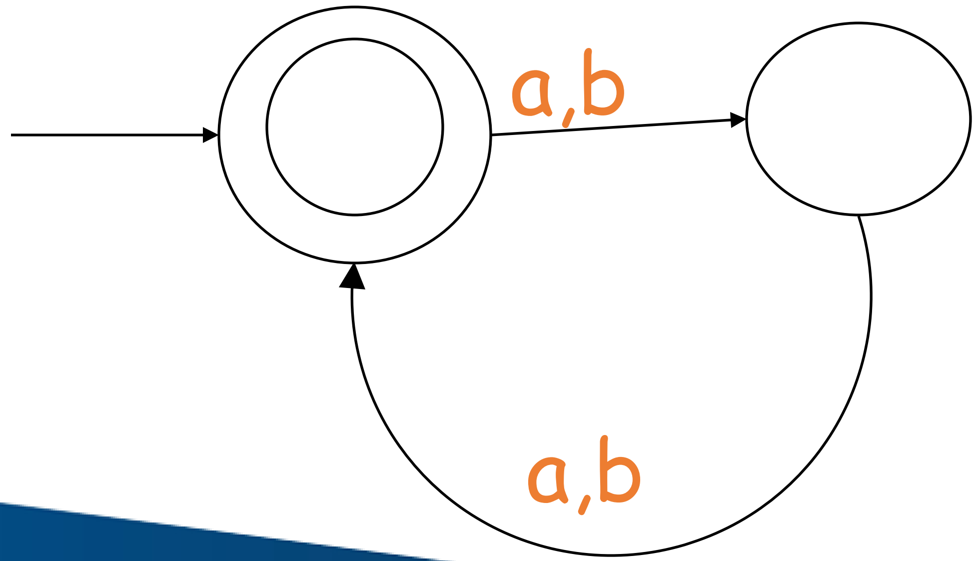
DFA- nb(w)mod4=1, ∑={ a,b}



DfA – n1(w)mod3=2, ∑={ 0,1}

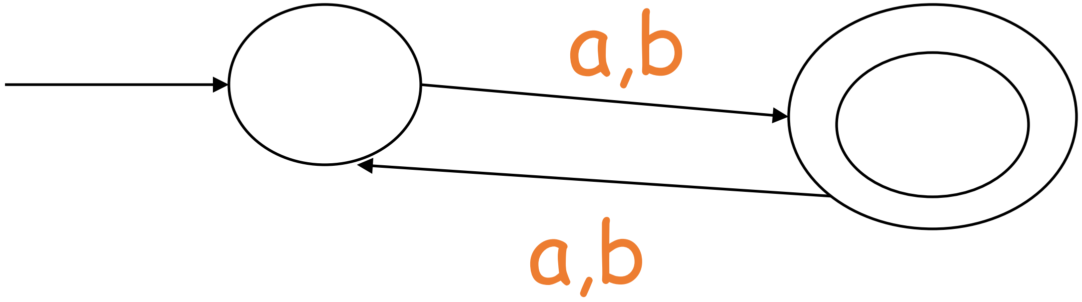


Dfa –lwlmod2=0,

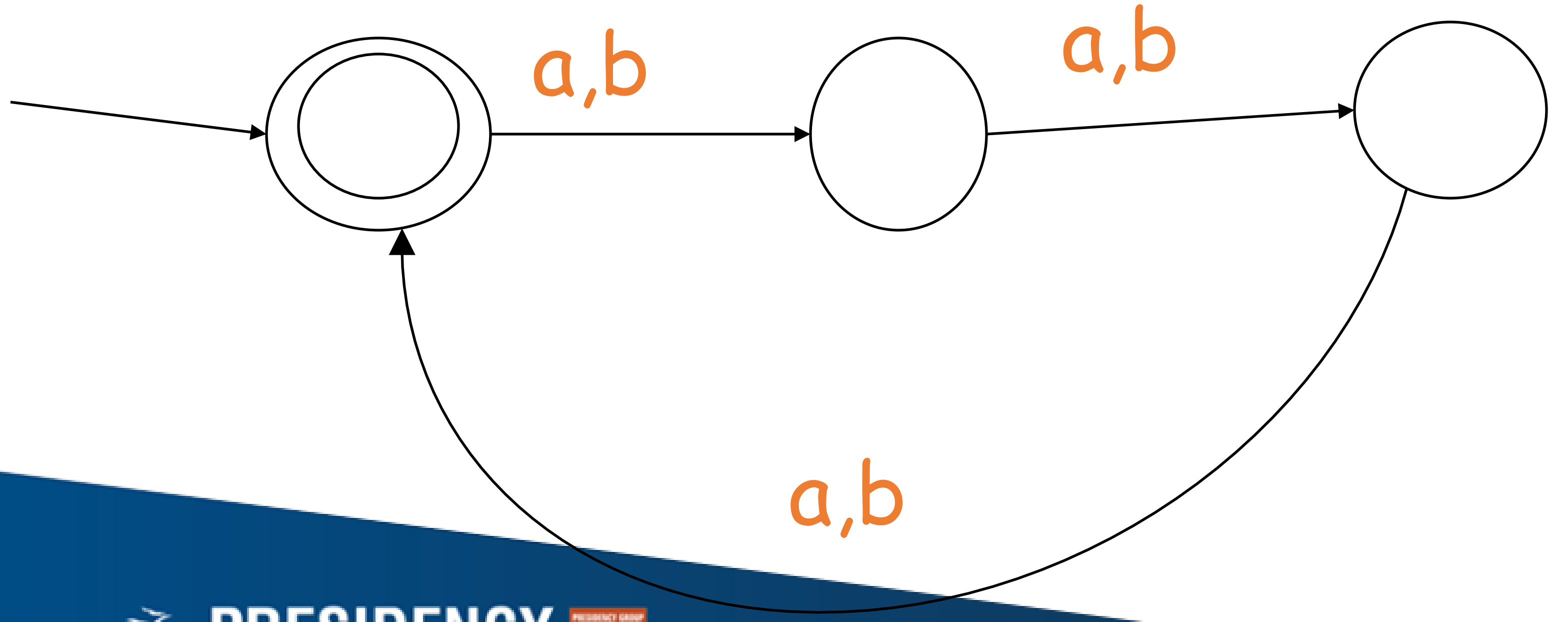


DfA – lwlmod2=1

L={a,b,aaa,aba,aab,baa,bbb,bab a,bba,....

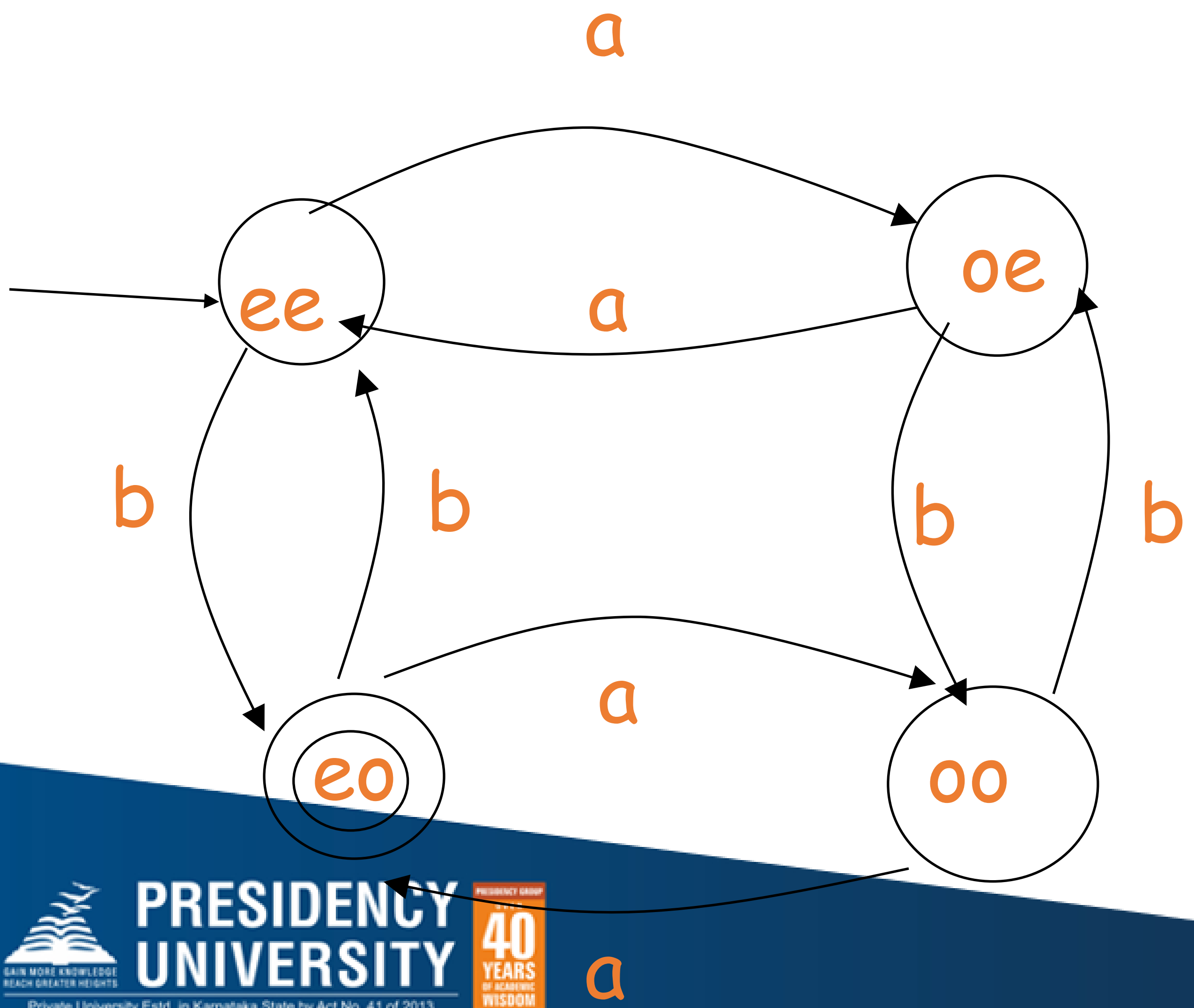


DFA –lwlmod3=0, ∑={ 0,1}

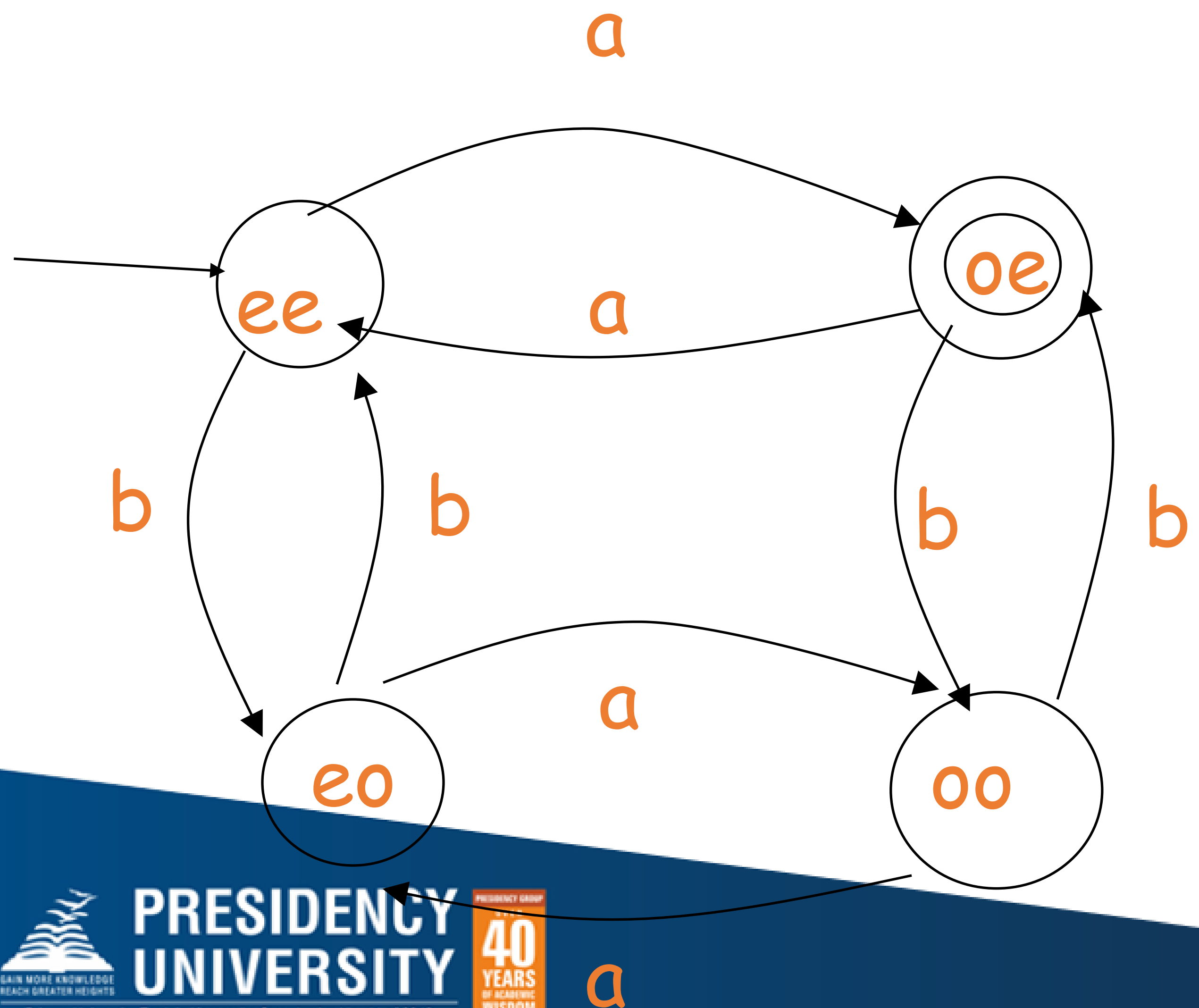


DFA : string with even number of 0’s & odd number of 1’s over {0, 1}

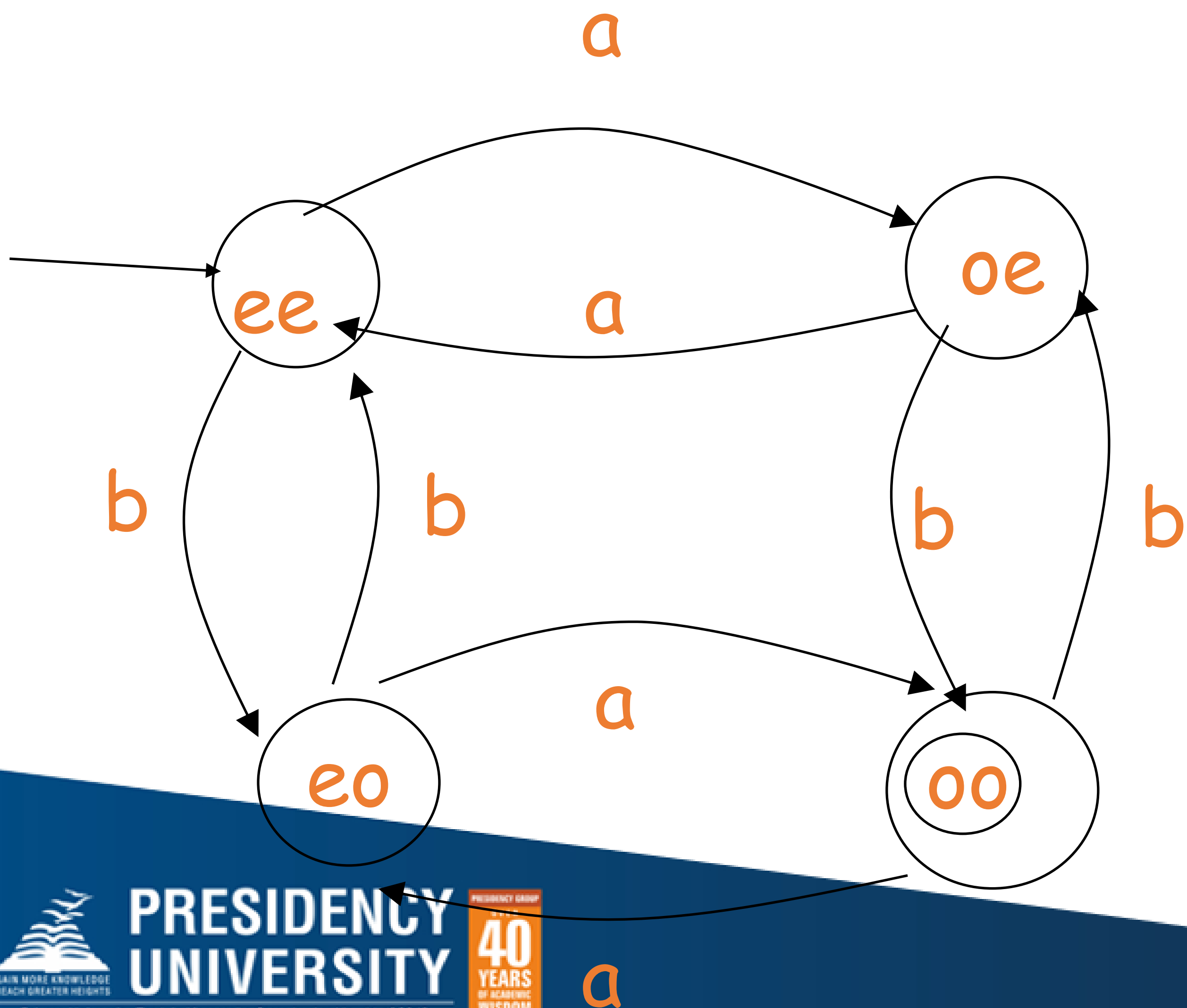
No of a's is even and no of b's odd-eo



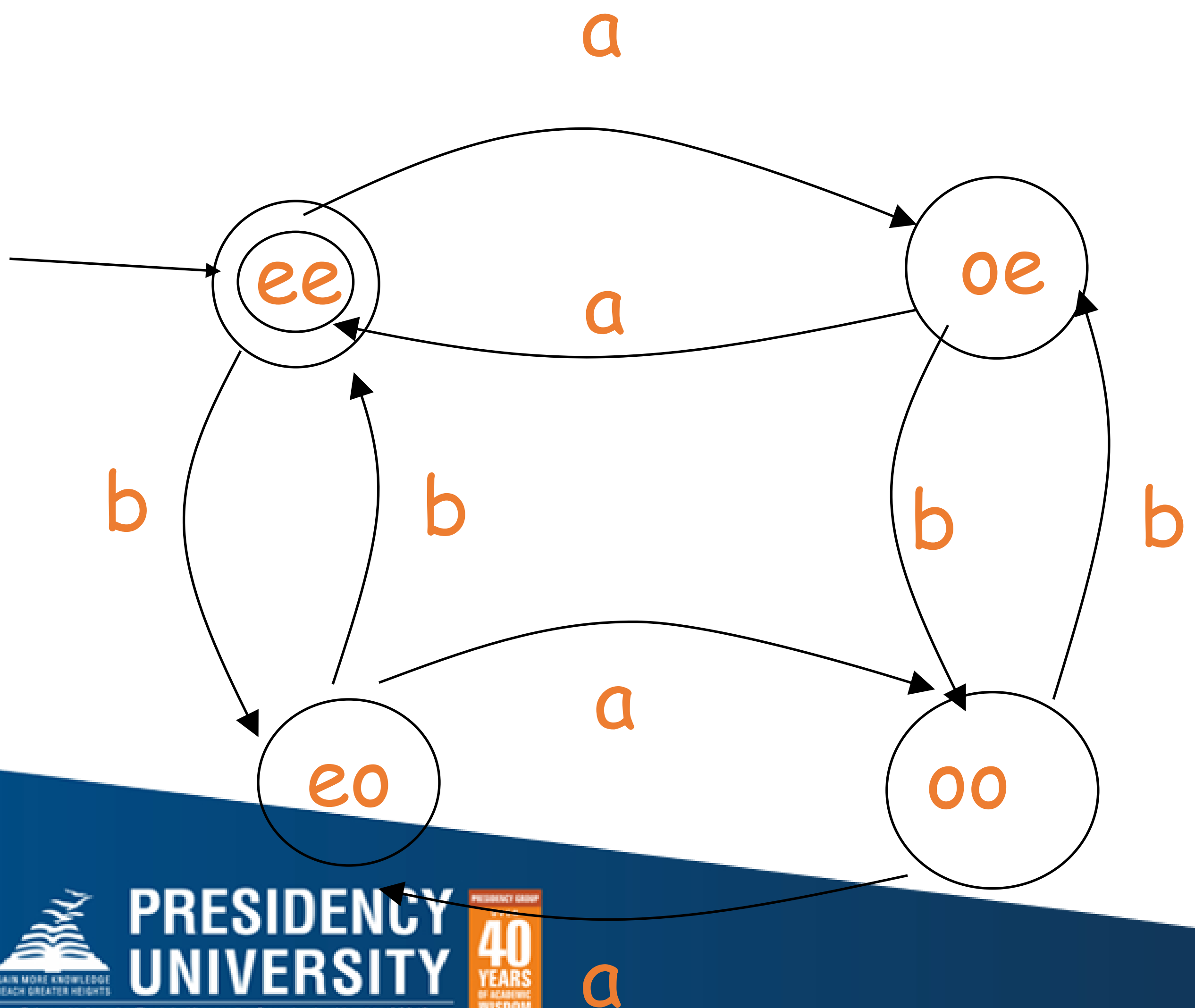
No of a's is odd and no of b's even=oe



No of a’s is odd and no of b’s odd-oo



No of a's is even and no of b's even-ee



NFA Definition

NFA is defined as $M=(Q, \sum, \delta, q0, F)$

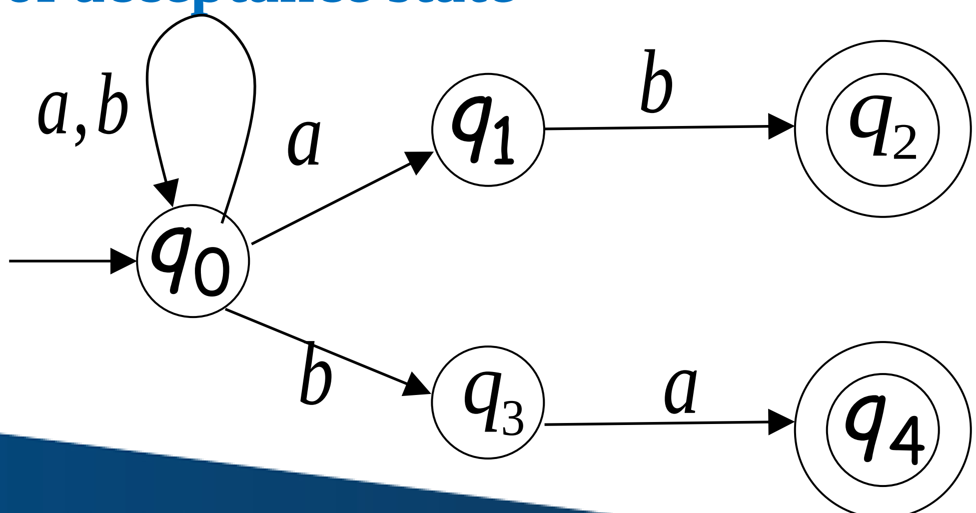
Where, Q= set of states

Σ = input alphabet

δ = transition function $\delta: Q \times (\Sigma \cup \{\epsilon\}) \rightarrow 2^Q$

q0→ Start / Initial State

F → set of acceptance state

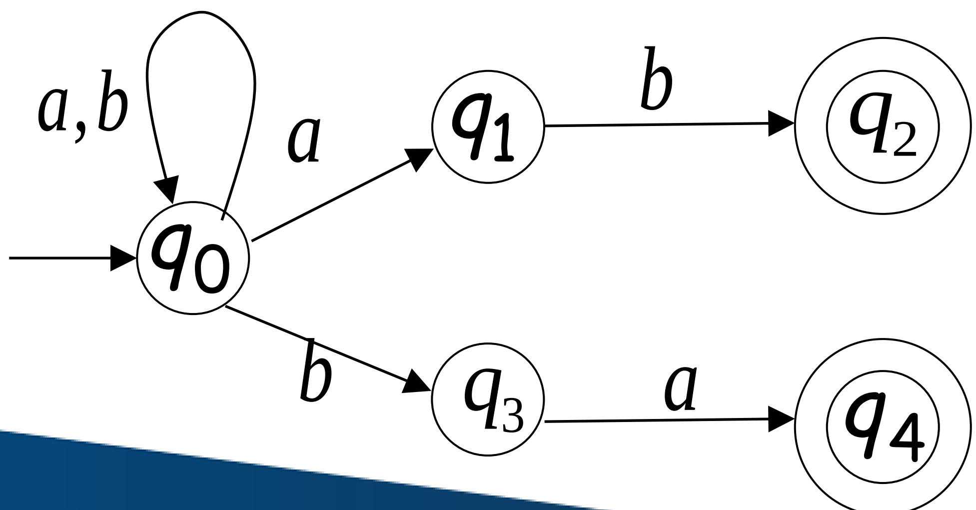


NFA Example

Design NFA that accepts language of strings ending with ab/ba over ∑ = {a, b}

Solution

- **Strings accepted= {ab, ba, abab, baba, bbab, aaaba, …}**
- **Strings rejected= {ϵ, a, b, aa, bb, aaa, abb, aaabbb, ...}**
- **Transition Diagram**

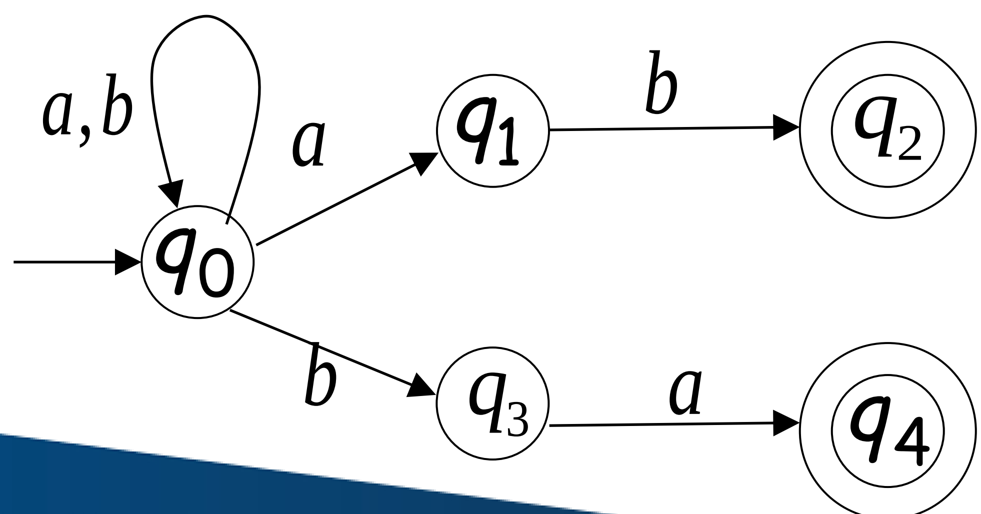


NFA to DFA Conversion Example

Convert NFA to DFA that accepts language of strings ending with ab/ba over ∑ = {a, b}

Solution

- **Draw NFA for given language**
- **Convert that NFA into DFA using subset construction method**



NFA to DFA Conversion Example

- Using Subset Construction Method
- Consider {q0} as Start State
- δ(q0, a)={q0 q1} newly formed state
- δ({q0 q1}, a)= δ(q0, a)u δ({q1, a})
  ={q0, q1} u {Ø}
  ={q0 q1}
- δ({q0 q1}, b)= δ(q0, b)u δ({q1, b})
  ={q0, q3} u {q2}
  ={q0 q2 q3}

NFA	a	b
q0	{q0, q1}	{q0, q3}
q1	-	{q2}
q2	-	-
q3	{q4}	-
q4	-	-

DFA	a	b
q0	{q0 q1}	{q0 q3}
{q0 q1}	{q0 q1}	{q0 q2 q3}

NFA to DFA Conversion Example

- Using Subset Construction Method
- Consider {q0} as Start State
- δ(q0, a)={q0 q1} newly formed state
- δ({q0 q1}, a)= δ(q0, a)u δ({q1, a})
  ={q0, q1} u {Ø}
  ={q0 q1}
- δ({q0 q1}, b)= δ(q0, b)u δ({q1, b})
  ={q0, q3} u {q2}
  ={q0 q2 q3}
- {q0 q2 q3} is newly formed state
- δ({q0 q2 q3}, a)= δ(q0, a) u δ({q2, a}) u δ({q3, a})
  ={q0, q1} u {Ø} u {q4}
  ={q0 q1 q4}
- δ({q0 q2 q3}, b)= δ(q0, b)u δ({q2, b}) u δ({q3, b})
  ={q0, q3} u {Ø} u {Ø} ={q0 q3}

NFA	a	b
q0	{q0, q1}	{q0, q3}
q1	-	{q2}
q2	-	-
q3	{q4}	-
q4	-	-

DFA	a	b
q0	{q0 q1}	{q0 q3}
{q0 q1}	{q0 q1}	{q0 q2 q3}
{q0 q2 q3)	{q0 q1 q4}	{q0 q3}

NFA to DFA Conversion Example

- {q0 q3} is newly formed state
- δ({q0 q3}, a)= δ(q0, a)∪ δ({q3, a})
  ={q0, q1} ∪ {q4}
  ={q0 q1 q4}
- δ({q0 q3}, b)= δ(q0, b)∪ δ({q3, b})
  ={q0, q3} ∪ {Ø}
  ={q0 q3}

NFA	a	b
q0	{q0, q1}	{q0, q3}
q1	-	{q2}
q2	-	-
q3	{q4}	-
q4	-	-

DFA	a	b
q0	{q0 q1}	{q0 q3}
{q0 q1}	{q0 q1}	{q0 q2 q3}
{q0 q2 q3)	{q0 q1 q4}	{q0 q3}
{q0 q3}	{q0 q1 q4}	{q0 q3}

NFA to DFA Conversion Example

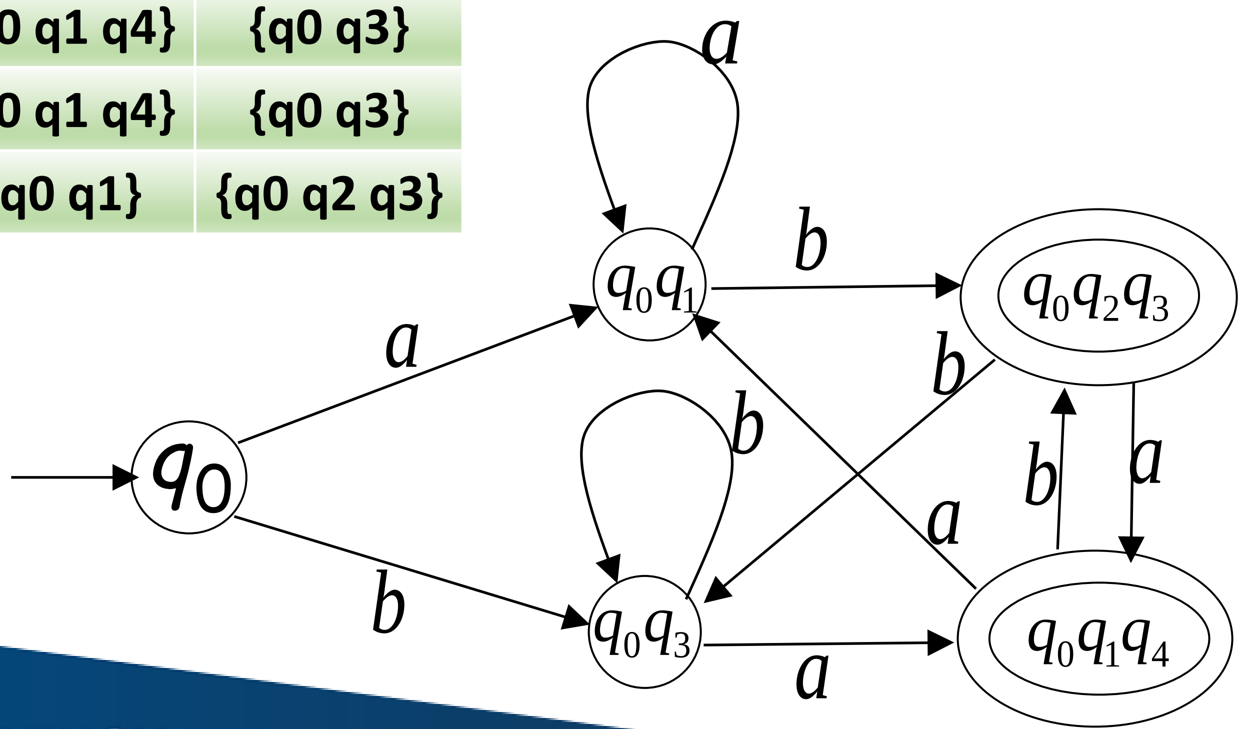
- {q0 q3} is newly formed state
- δ({q0 q3}, a)= δ(q0, a)∪ δ({q3, a})
  ={q0, q1} ∪ {q4}
  ={q0 q1 q4}
- δ({q0 q3}, b)= δ(q0, b)∪ δ({q3, b})
  ={q0, q3} ∪ {Ø}
  ={q0 q3}
- {q0 q1 q4} is newly formed state
- δ({q0 q1 q4}, a)= δ(q0, a) ∪ δ({q1, a}) ∪ δ({q4, a})
  ={q0, q1} ∪ {Ø} ∪ {Ø}
  ={q0 q1}
- δ({q0 q1 q4}, b)= δ(q0, b)∪ δ({q1, b}) ∪ δ({q4, b})
  ={q0, q3} ∪ {q2} ∪ {Ø}
  ={q0 q2 q3}

NFA	a	b
q0	{q0, q1}	{q0, q3}
q1	-	{q2}
q2	-	-
q3	{q4}	-
q4	-	-

DFA	a	b
q0	{q0 q1}	{q0 q3}
{q0 q1}	{q0 q1}	{q0 q2 q3}
{q0 q2 q3)	{q0 q1 q4}	{q0 q3}
{q0 q3}	{q0 q1 q4}	{q0 q3}
{q0 q1 q4}	{q0 q1}	{q0 q2 q3}

NFA to DFA Conversion Example

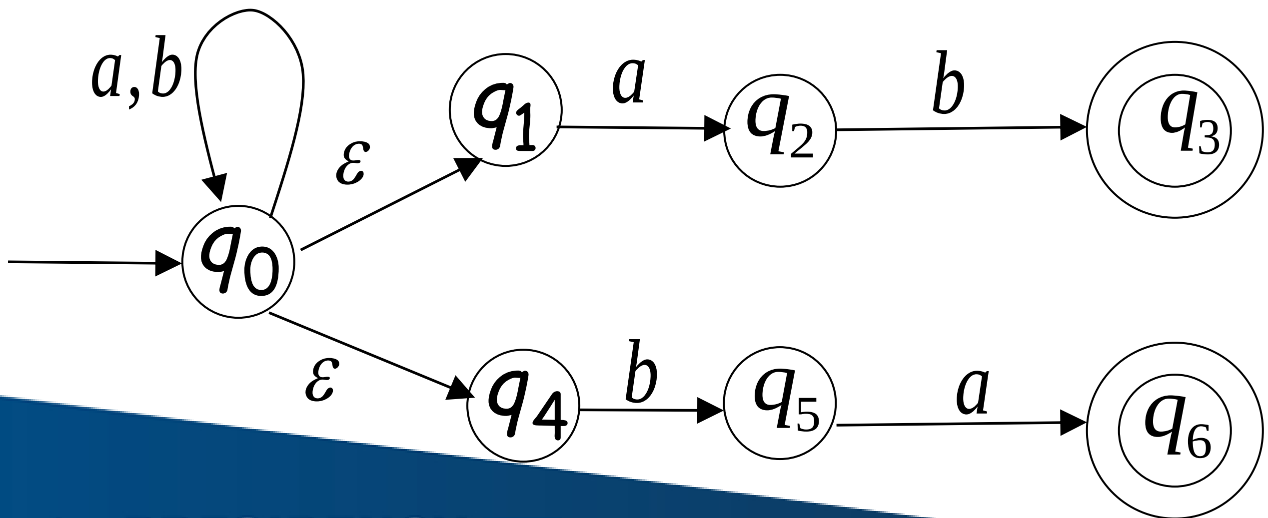
DFA	a	b
q0	{q0 q1}	{q0 q3}
{q0 q1}	{q0 q1}	{q0 q2 q3}
{q0 q2 q3)	{q0 q1 q4}	{q0 q3}
{q0 q3}	{q0 q1 q4}	{q0 q3}
{q0 q1 q4}	{q0 q1}	{q0 q2 q3}



ϵ-NFA Example

ϵ- NFA allows transition on null string or no inputs

It means machine can make transition without input

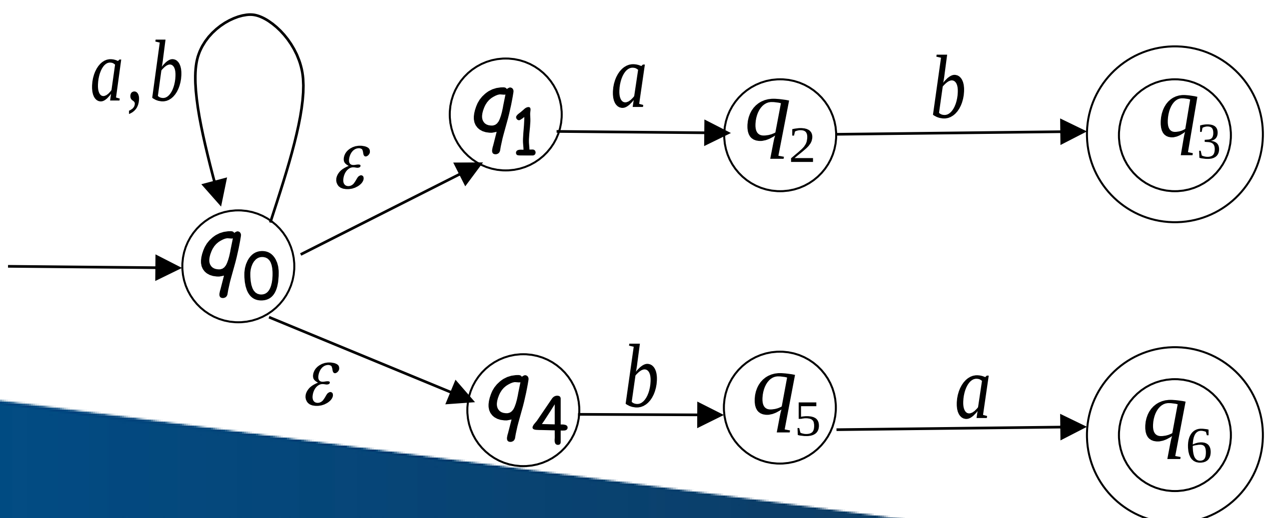


ϵ-NFA Example

Design ϵ-NFA that accepts language of strings ending with ab/ba over ∑ = {a, b}

Solution

- **Strings accepted= {ab, ba, abab, baba, bbab, aaaba, …}**
- **Strings rejected= {ϵ, a, b, aa, bb, aaa, abb, aaabbb, ...}**
- **Transition Diagram**

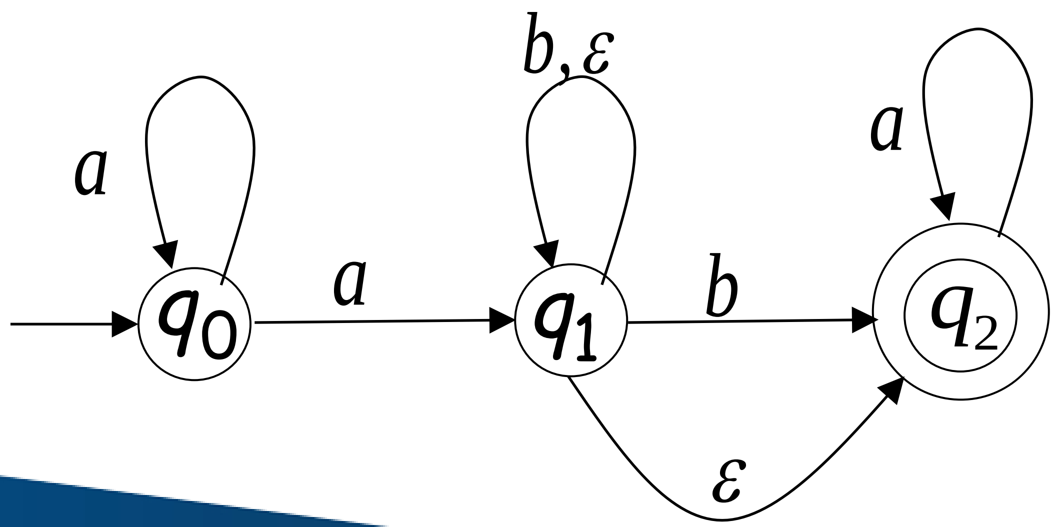


ϵ-NFA to DFA Conversion Example

Convert the following ϵ-NFA to DFA

Solution

- **Find ϵ-closure of each state**
- **Convert that ϵ-NFA into DFA using subset construction method**



ϵ-NFA to DFA Conversion Example

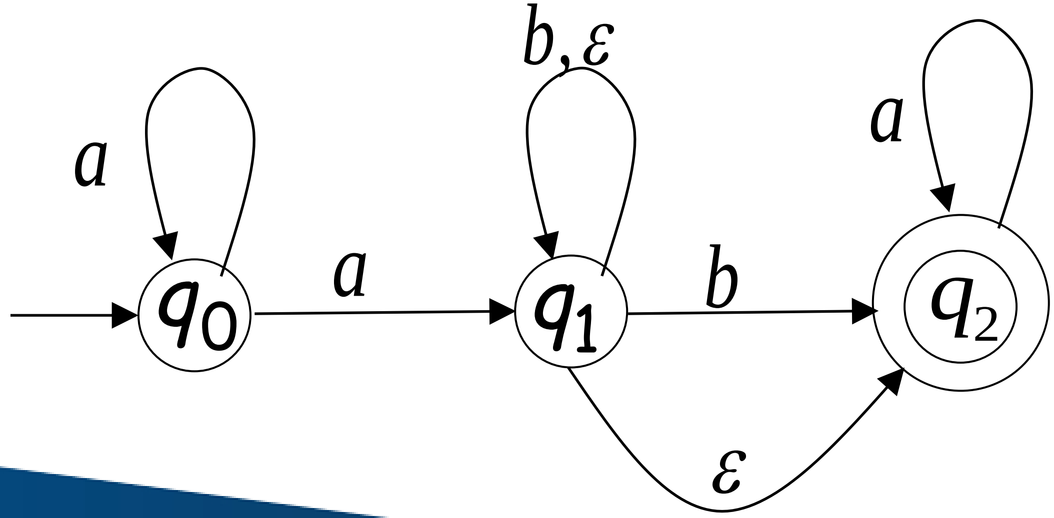
ϵ- Closure : of a state qi is the set of states including qi, also states having ϵ-transitions from state qi and so on…

ϵ- Closure of state qi includes:

-qi

-set of states reachable from qi over ϵ- move

-set of states reachable from states of state qi over ϵ- move and so on…



ϵ-NFA to DFA Conversion Example

- ϵ-closure of state is define as a set of states that includes
- that state itself
- set of states reachable from state Q over ϵ-transaction
- set of states reachable from existing states in ϵ-closure over ϵ-transaction and so on..

States	ϵ-closure
q0	{q0}
q1	{q1, q2}
q2	{q2}

ϵ-NFA to DFA Conversion Example

- Using Subset Construction Method
- Consider ϵ-closure of {q0}= {q0} as Start State
- a-successor of {q0}
  = ϵ-closure (δ(q0, a))
  = ϵ-closure ({q0,q1})
  = ϵ-closure (q0) ∪ ϵ-closure (q1)
  = {q0} ∪ {q1, q2}
  ={q0,q1,q2}
- b-successor of {q0}
  = ϵ-closure (δ(q0, b))={Ø}

DFA	a	b
{q0}	{q0 q1 q2}	{Ø}

ϵ-NFA to DFA Conversion Example

- Using Subset Construction Method
- Consider {q0 q1 q2} as a new state
- a-successor of {q0 q1 q2}

  = ϵ-closure (δ(q0, a)∪δ(q1, a)∪δ(q2, a))

  = ϵ-closure ({q0 q1}∪{Ø}∪{q2})

  = ϵ-closure ({q0 q1 q2})

  = ϵ-closure (q0) ∪ ϵ-closure (q1) ∪ ϵ-closure (q2)

  = {q0} ∪ {q0, q1} ∪ {q2}={q0,q1,q2}
- b-successor of {q0 q1 q2}

  ϵ-closure (δ(q0, b)∪δ(q1, b)∪δ(q2, b))

  = ϵ-closure ({Ø}∪{q1 q2}∪{Ø})

  = ϵ-closure ({q1 q2})

  = ϵ-closure (q1) ∪ ϵ-closure (q2) = {q1 q2} ∪ {q2}={q1 q2}

DFA	a	b
{q0}	{q0 q1 q2}	{Ø}
{q0 q1 q2}	{q0 q1 q2}	{q1 q2}

ϵ-NFA to DFA Conversion Example

- **Using Subset Construction Method**
- **Consider {q1 q2} as a new state**
- **a-successor of {q1 q2}**

  = ϵ-closure (δ(q1, a)∪δ(q2, a))

  = ϵ-closure ({Ø}∪{q2})

  = ϵ-closure ({q2})

  = {q2}
- **b-successor of {q1 q2}**

  ϵ-closure (δ(q1, b)∪δ(q2, b))

  = ϵ-closure ({q1 q2}∪{Ø})

  = ϵ-closure ({q1 q2})

  = ϵ-closure (q1) ∪ ϵ-closure (q2) = {q1 q2} ∪ {q2}={q1 q2}

DFA	a	b
{q0}	{q0 q1 q2}	{Ø}
{q0 q1 q2}	{q0 q1 q2}	{q1 q2}
{q1 q2)	{q2}	{q1 q2}

ϵ-NFA to DFA Conversion Example

- Using Subset Construction Method
- Consider {q2} as a new state
- a-successor of {q2}

  = ϵ-closure (δ(q2, a))

  = ϵ-closure ({q2})

  = {q2}
- b-successor of {q2}

  =ϵ-closure (δ(q2, b))

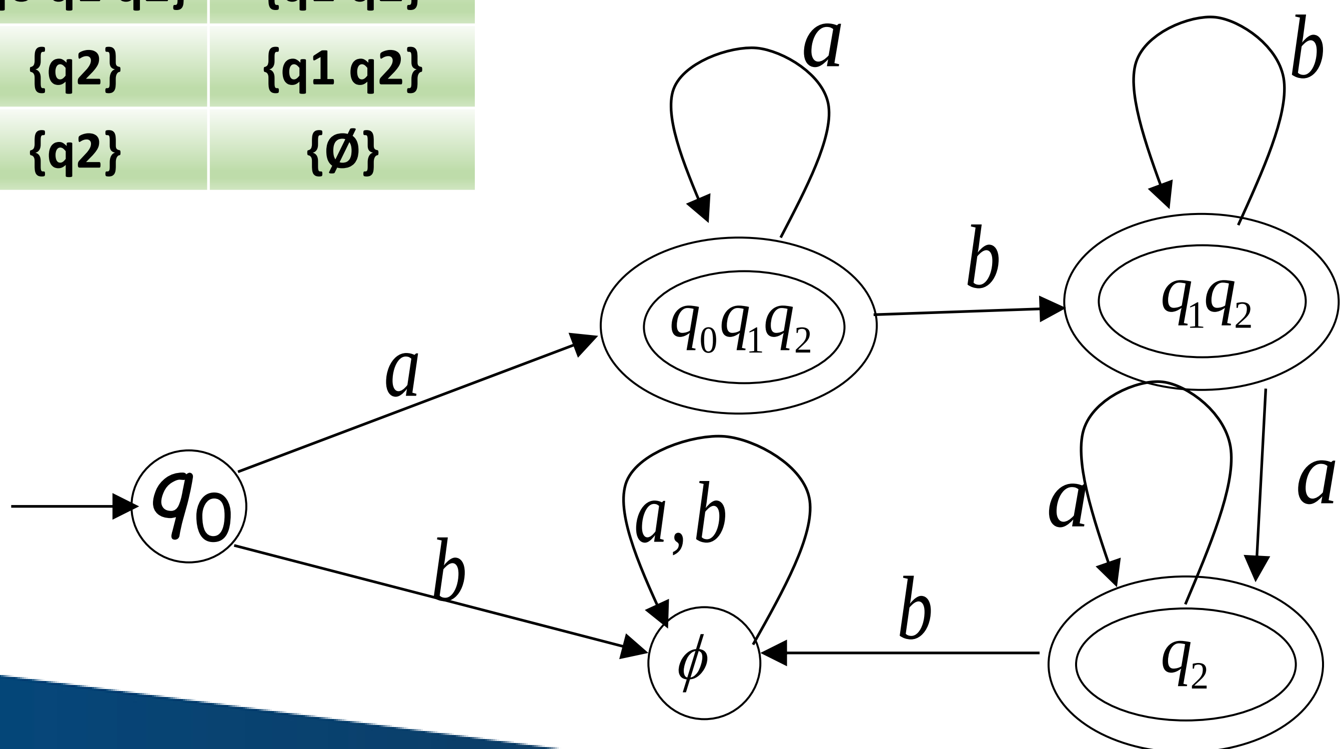
  = ϵ-closure ({Ø})

  ={Ø}

DFA	a	b
{q0}	{q0 q1 q2}	{Ø}
{q0 q1 q2}	{q0 q1 q2}	{q1 q2}
{q1 q2)	{q2}	{q1 q2}
{q2}	{q2}	{Ø}

ϵ-NFA to DFA Conversion Example

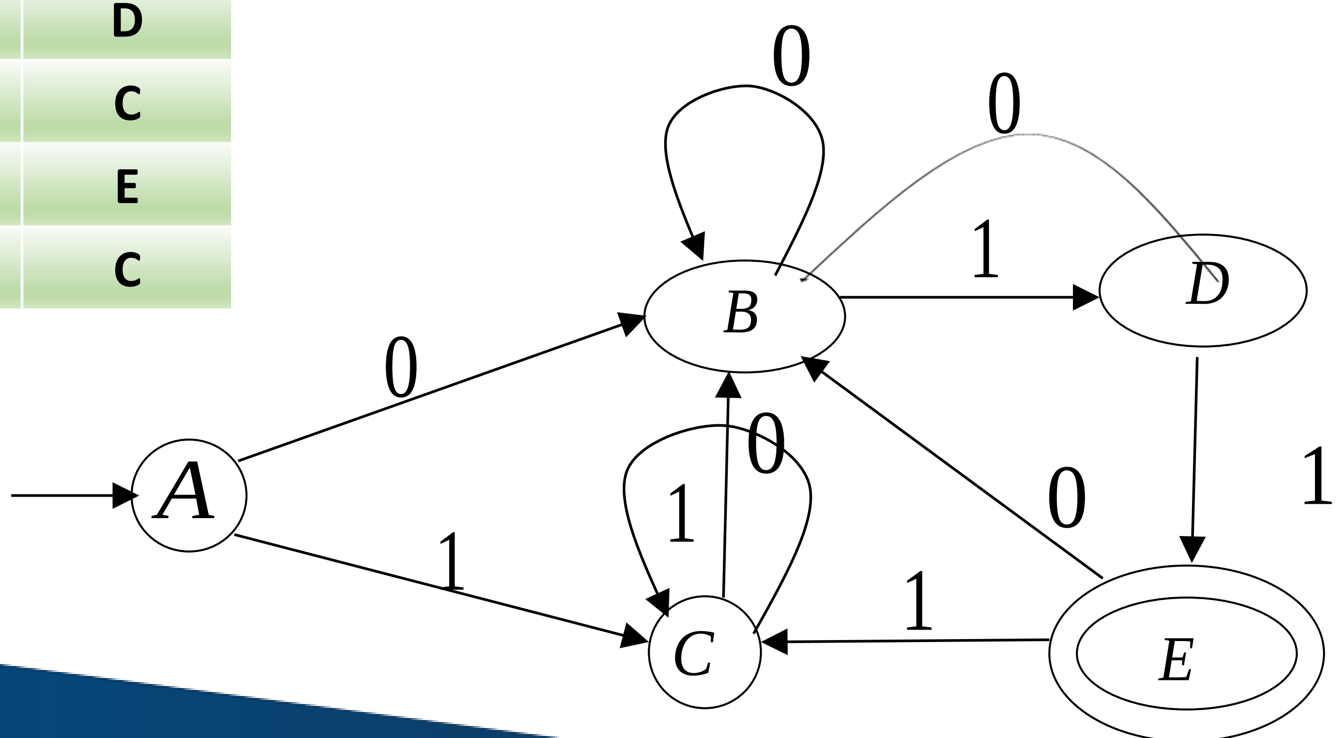
DFA	a	b
{q0}	{q0 q1 q2}	{Ø}
{q0 q1 q2}	{q0 q1 q2}	{q1 q2}
{q1 q2)	{q2}	{q1 q2}
{q2}	{q2}	{Ø}



Minimization of DFA Example

Minimize the following DFA

DFA	0	1
→A	B	C
B	B	D
C	B	C
D	B	E
E*	B	C



Minimization of DFA Example

Minimize the following DFA

- Use State Equivalence Method
- Write 0'Equivalence as

  {A, B, C, D}   {E}
- Write 1'Equivalence

  {A, B, C} {D} {E}
- Write 2'Equivalence

  {A, C} {B} {D} {E}
- Write 3'Equivalence

  {A, C} {B} {D} {E}

DFA	0	1
→A	B	C
B	B	D
C	B	C
D	B	E
E*	B	C

PRESIDENCY UNIVERSITY
GAIN MORE KNOWLEDGE REACH GREATER HEIGHTS
Private University Estd. in Karnataka State by Act No. 41 of 2013
PRESIDENCY GROUP OVER 40 YEARS OF ACADEMIC WISDOM

Minimization of DFA Example

Minimize the following DFA

- **Use State Equivalence Method**
- **Write 0'Equivalence as**

  {A, B, C, D} {E}
- **Write 1'Equivalence**

  {A, B, C} {D} {E}
- **Write 2'Equivalence**

  {A, C} {B} {D} {E}
- **Write 3'Equivalence**

  {A, C} {B} {D} {E}

DFA	0	1
→A,C	B	C
B	B	D
D	B	E
E*	B	C