

# Pune Institute of Computer Technology, Pune

**COVID19**

## **STAY SAFE, STAY HEALTHY.**

You can reduce the risk of spreading the coronavirus that causes COVID-19 by taking the same steps you'd take to avoid getting colds.



### **LET'S ALL DO OUR PART**

- 1  Wash hands frequently with soap
- 2  Wear a mask if you have a cough or runny nose
- 3  Cover your mouth with a tissue paper when coughing or sneezing
- 4  See a doctor if you feel unwell

# **THEORY OF COMPUTATION**

**Jagadish Kashinath Kamble** (M.Tech, IIT Kharagpur)

Dept. of Information Technology,  
Pune Institute of Computer Technology, Pune

# KYC: Know Your Course

- Course Instructor: Jagadish Kashinath Kamble
- A3-310, jkkamble@pict.edu
- Slides will be made available at PICT Website and Microsoft Teams Class Group.
- Books
  - 1. Introduction to the Theory of Computation by Michael Sipser
  - 2. Theory of Computation, Vivek Kulkarni, Oxford University Press
  - 3. John C. Martin, Introduction to Language and Theory of Computation
  - 4. Hopcroft Ulman, Introduction to Automata Theory, Languages and Computations

Many other books are available and may serve the same purpose

# KYC: Know Your Course

- Online Lectures / Classroom Lectures
- Evaluation in the theory course (100 Marks):

In-Semester-30%

End-semester -70%

- **Attendance REALLY matters**
- Important for understanding the course
- **Classes will involve both Slides + Board (to roughly equal degrees)**
- .

# Pre-requisites

- Data Structures
- Discrete Structures

# Important Dates(Tentative)

- Class Test 1: 3<sup>rd</sup> Week of Sep-2021
- In-Semester : 1<sup>st</sup> Week of Oct-2021
- Class Test 2: 3<sup>rd</sup> Week of Nov-2021
- End-semester: Dec/Jan-2020-21

(The exact Time will be announced later)

# Program Outcomes

**Programme Outcomes** are narrow statements that describe what the students are expected to know and would be able to do upon the graduation. These relate to the skills, knowledge, and behavior that students acquire through the programme.

The programme outcomes adopted by NBA for accreditation of programme are based on initial capabilities, competence, skills, etc. These parameters are called Graduates Attributes and they vary from discipline to discipline and level to level.

# Program Outcomes

Engineering Graduates will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

# Program Outcomes

4. **Conduct investigation of Complex Problems:** Use research based knowledge and research methods including design experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

# Program Outcomes

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project Management and Finance:** Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

# Course Objectives

The learning objectives of this course are to

1. To understand problem classification and problem solving by machines.
2. To understand the basics of automata theory and its operations.
3. To study computing machines by describing, classifying and comparing different types of computational models.
4. Encourage students to study theory of computability and complexity.
5. To understand the P and NP class problems and its classification.
6. To understand the fundamentals of problem decidability and reducibility.

# Course Outcomes

- **Course Outcomes** are narrower statements that describe what students are expected to know, and be able to do at the end of each course/subject. While the POs define the departmental outcomes, the COs are more oriented towards the subjects and are mostly defined by the faculties consulting higher authorities.
- The COs are more like statements that relate to the skills, knowledge, and behavior the students acquire as they go through a specific course within a program. They collectively contribute to the program outcomes. They are to be mapped to the POs, and not necessarily to a single one.
- Two or more COs can be mapped to a PO and a CO can be mapped to one or more PO(s). COs are mapped to different POs based on their influence on them.

# Learning/Course Outcomes

After completing this course, students will be able to:

1. To construct finite state machines to solve problems in computing.
2. To write mathematical expressions for the formal languages
3. To apply and construct well defined rules for syntax verification.
4. To construct and analyze Push Down, Post and Turing Machine for formal languages.
5. To express the understanding of the decidability and undecidability and computational complexity.

# Program Educational Objectives (PEOs)

- The educational objectives of an engineering degree program are the statements that describe the expected achievements of graduates in their career, and what the graduates are expected to perform and achieve during the first few years after graduation.
- Program Educational Objectives Essentially Answer the Question: Why does the Program exist in the first place?
- *PEOs are:*
  - Statement of areas or fields where the graduates find employment
  - Preparedness of graduates to take up higher studies

# Program Educational Objectives (PEOs)

- **PEO1.** To produce graduates who would have developed a strong background in basic science and mathematics and ability to use these tools in their chosen fields of specialization.
- **PEO2.** To produce graduates who have the ability to demonstrate technical competence in the fields of information Technology and develop solutions to the problems.
- **PEO3.** To produce graduates who would attain professional competence through life-long learning such as advanced degrees, professional registration, and other professional activities.
- **PEO4.** To produce graduates who function effectively in a multi-disciplinary environment and individually, within a global, societal, and environmental context.
- **PEO5.** To produce graduates with ethical and moral behavior

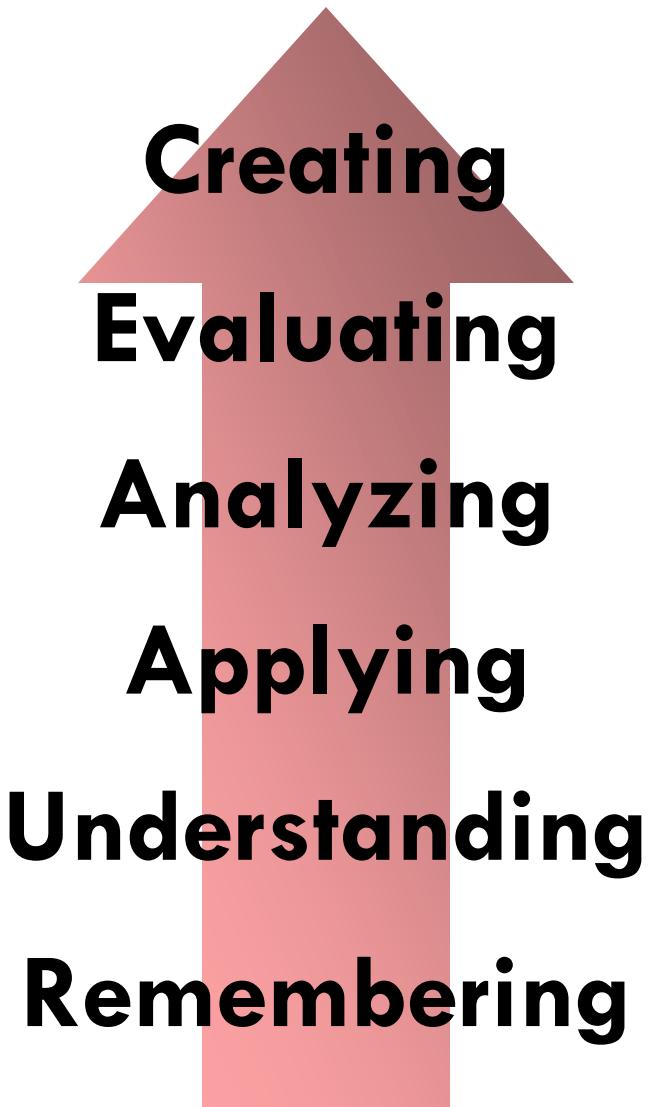
# Bloom's Taxonomy

**Taxonomy = Classification**

**Classification of thinking**

**Six cognitive levels of complexity**

# Bloom's Taxonomy



**Creating**

**Evaluating**

**Analyzing**

**Applying**

**Understanding**

**Remembering**

# Remembering

**The learner is able to recall, restate and remember learned information**

Describing  
Finding  
Identifying  
Listing

Retrieving  
Naming  
Locating  
Recognizing



*Can students recall information?*

# Understanding

**Student grasps meaning of information by interpreting and translating what has been learned**

Classifying  
Comparing  
Exemplifying  
Explaining



Inferring  
Interpreting  
Paraphrasing  
Summarizing

*Can students explain ideas or concepts?*

# Applying

**Student makes use of information in a context  
different from the one in which it was learned**

Implementing  
Carrying out



Using  
Executing

*Can students use the information in  
another familiar situation?*

# Analysing

**Student breaks learned information into its parts to best understand that information**

- Attributing
- Comparing
- Deconstructing
- Finding



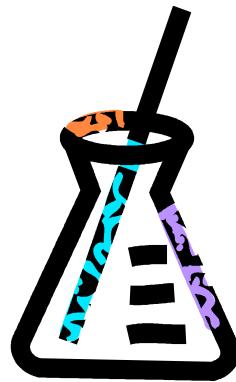
- Integrating
- Organizing
- Outlining
- Structuring

*Can students break information into parts to explore understandings and relationships?*

# Evaluating

**Student makes decisions based on in-depth reflection, criticism and assessment**

Checking  
Critiquing  
Detecting  
Experimenting



Hypothesising  
Judging  
Monitoring  
Testing

*Can students justify a decision or a course of action?*

# Creating

**Student creates new ideas and information using what previously has been learned**

Constructing  
Designing  
Devising  
Inventing

Making  
Planning  
Producing



*Can students generate new products, ideas, or ways of viewing things?*

# **Course Contents**



**Unit I. FINITE STATE MACHINES**

**Unit II. REGULAR EXPRESSIONS**

**Unit III. CONTEXT FREE GRAMMAR AND LANGUAGES**

**Unit IV. PUSHDOWN AUTOMATA AND POST MACHINES**

**Unit V. TURING MACHINES**

**Unit VI.COMPUTATIONAL COMPLEXITY**

# Course Contents

## UNIT - I FINITE STATE MACHINES

**Basic Concepts:** Symbols, Strings, Language, Formal Language, Natural Language. Basic Machine and Finite State Machine.

**FSM without output:** Definition and Construction-DFA, NFA, NFA with epsilon-Moves, Minimization Of FA, Equivalence of NFA and DFA, Conversion of NFA with epsilon moves to NFA, Conversion of NFA With epsilon moves to DFA.

**FSM with output:** Definition and Construction of Moore and Mealy Machines, Inter-conversion between Moore and Mealy Machines.

## UNIT - II REGULAR EXPRESSIONS

Definition and Identities of Regular Expressions, Construction of Regular Expression of the given L,

Construction of Language from the RE, Construction of FA from the given RE using direct method, Conversion of FA to RE using Arden's Theorem, Pumping Lemma for RL, Closure properties of RLs, Applications of

# Course Contents

## UNIT - III CONTEXT FREE GRAMMAR AND LANGUAGES

Introduction, Formal Definition of Grammar, Notations, Derivation Process: Leftmost Derivation, Rightmost Derivation, derivation trees, Context Free Languages, Ambiguous CFG, Removal of ambiguity, Simplification of CFG, Normal Forms, Chomsky Hierarchy, Regular grammar, equivalence of RG(LRG and RLG) and FA.

## UNIT IV PUSHDOWN AUTOMATA AND POST MACHINES

**Push Down Automata:** Introduction and Definition of PDA, Construction (Pictorial/ Transition diagram) of PDA, Instantaneous Description and ACCEPTANCE of CFL by empty stack and final state, Deterministic PDA Vs Nondeterministic PDA, Closure properties of CFLs, pumping lemma for CFL.

**Post Machine-** Definition and construction.

# Course Contents

## UNIT - V TURING MACHINES

Formal definition of a Turing machine, Recursive Languages and Recursively Enumerable Languages, Design of Turing machines, Variants of Turing Machines: Multi-tape Turing machines, Universal Turing Machine, Nondeterministic Turing machines. Comparisons of all automata.

## UNIT - VI COMPUTATIONAL COMPLEXITY

**Decidability:** Decidable problems concerning regular languages, Decidable problems concerning context-free languages, Un-decidability, Halting Problem of TM, A Turing-unrecognizable language.

**Reducibility:** Un-decidable Problems from Language Theory, A Simple Undecidable Problem PCP, Mapping Reducibility

**Time Complexity:** Measuring Complexity, The Class P, Examples of problems in P, The Class NP, Examples of problems in NP, NP-completeness.

# THEORY OF COMPUTATION

---

Unit I

## FINITE STATE MACHINES

---

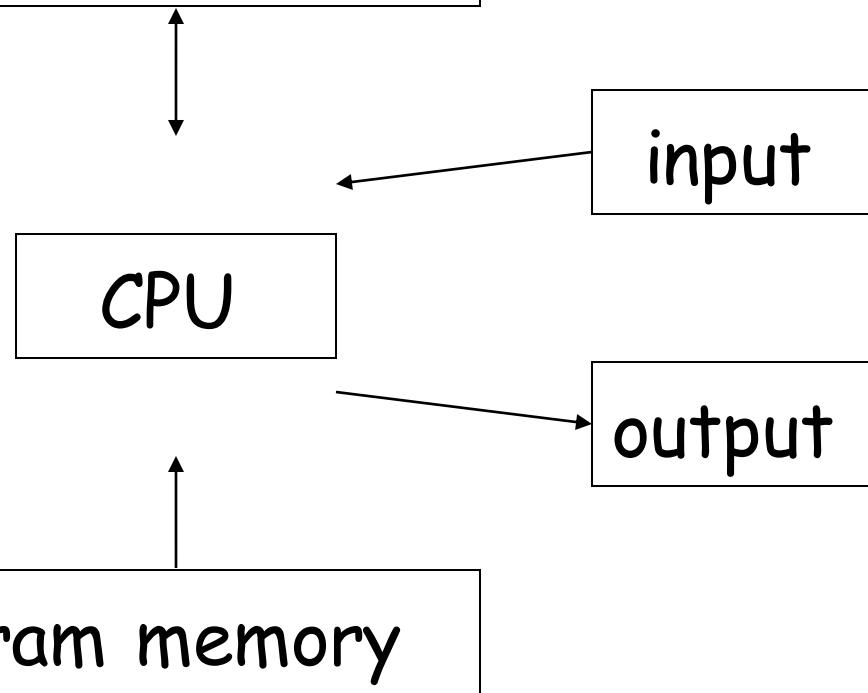
**Jagadish Kashinath Kamble** (M.Tech, IIT Kharagpur)

Dept. of Information Technology,  
Pune Institute of Computer Technology, Pune

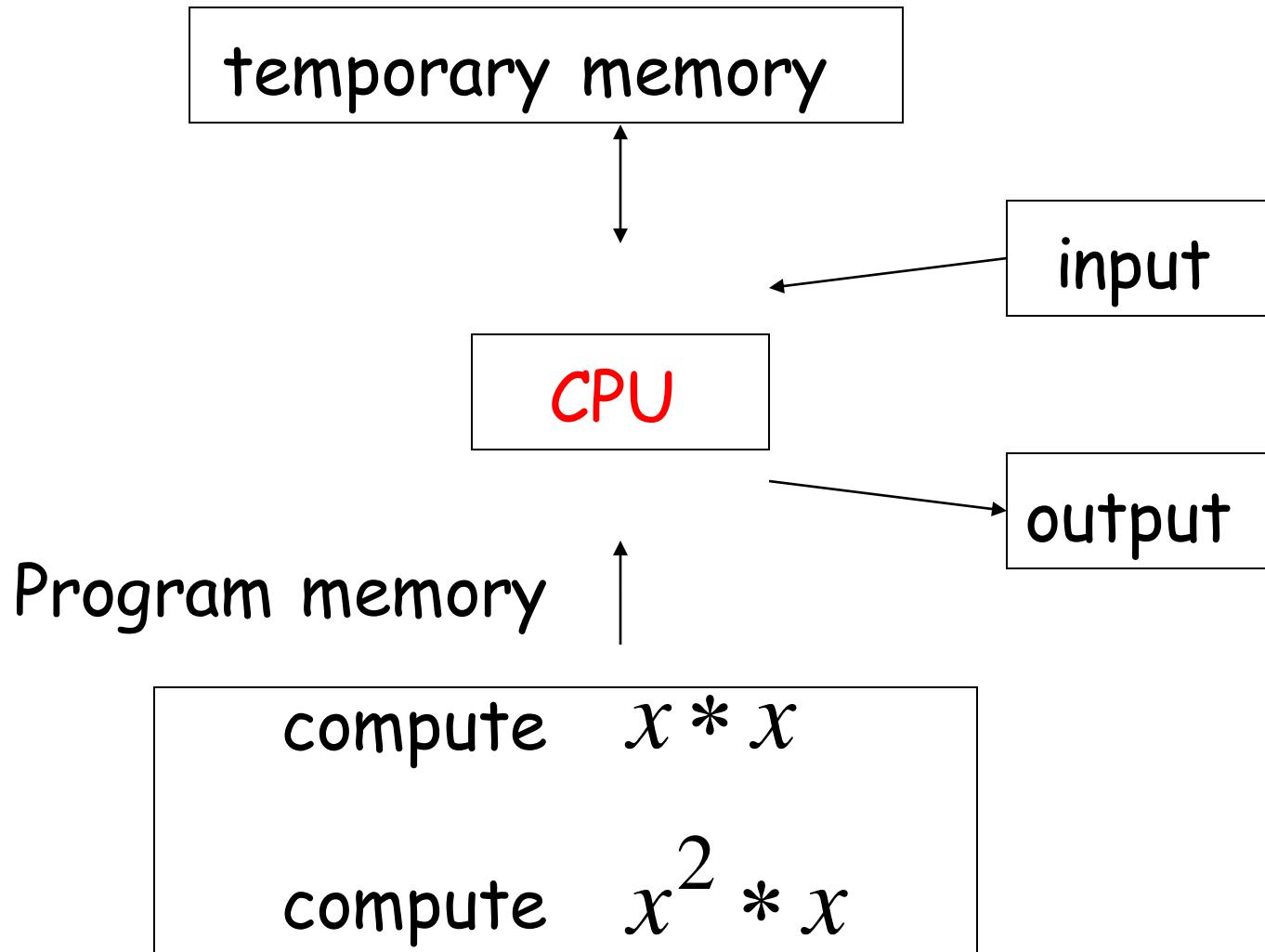
# Introduction

- Computations are designed to solve problems.
- Computational Devices
- Programs are descriptions of computations written for execution on computers.
- The field of computer science is concerned with the **development of methodologies for designing programs**, and with the development of computers for executing programs.
- It is therefore of **central importance** for those involved in the field that the characteristics of problems, programs, computation and computers be fully understood.

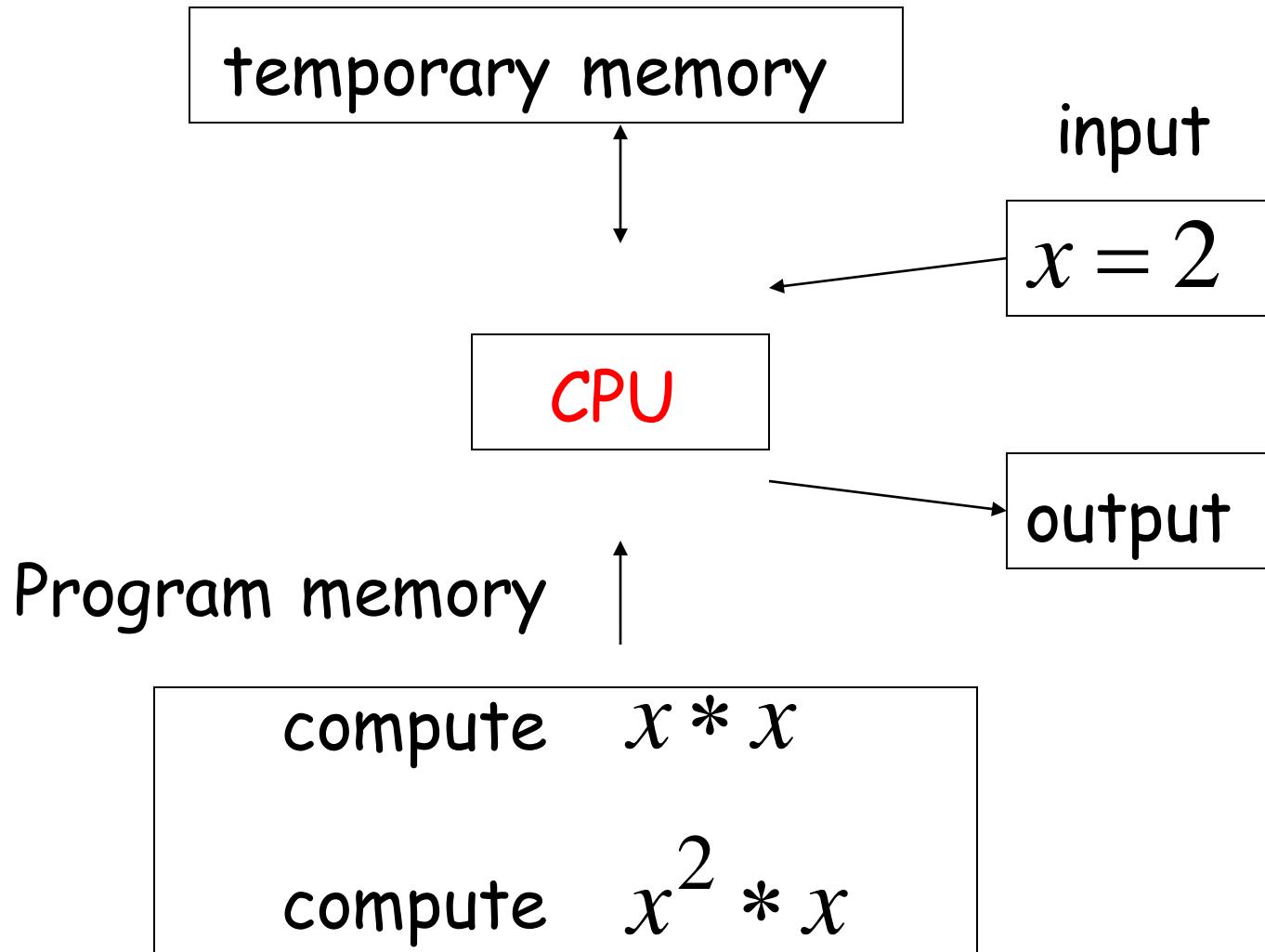
temporary memory



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



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



temporary memory

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

$$z = 2 * 2 = 4$$

$$f(x) = z * 2 = 8$$

input

$$x = 2$$

CPU

Program memory

output

$$\text{compute } x * x$$

$$\text{compute } x^2 * x$$

temporary memory

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

$$z = 2 * 2 = 4$$

$$f(x) = z * 2 = 8$$

input

$$x = 2$$

CPU

Program memory

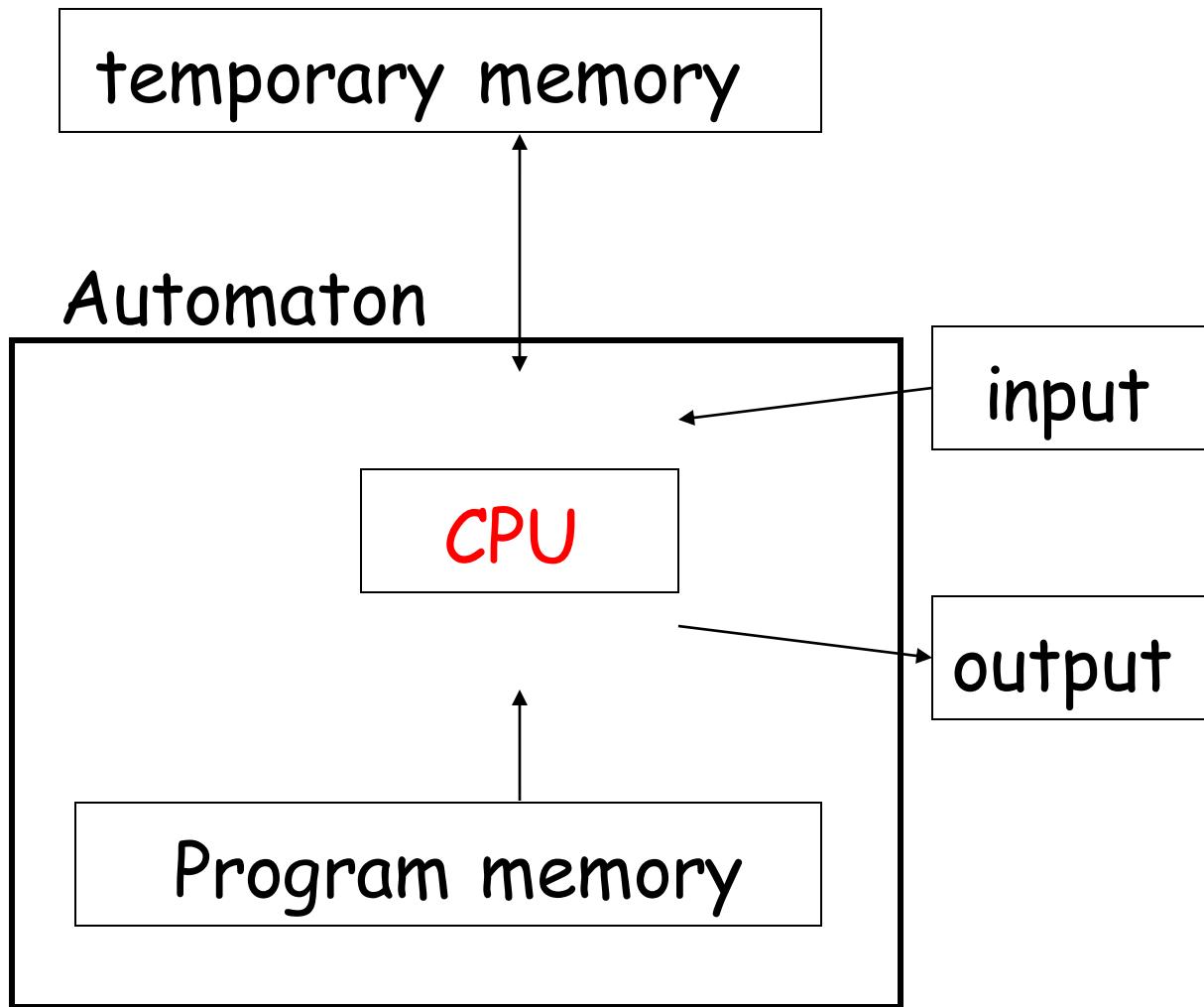
$$f(x) = 8$$

output

$$\text{compute } x * x$$

$$\text{compute } x^2 * x$$

# Automaton



# What is Automata Theory?

- *Study of abstract computing devices, or “machines”*
- **Automaton = an abstract computing device**
  - Note: A “device” need not even be a physical hardware!
- **A fundamental question in computer science:**
  - Find out what different models of machines can do and cannot do
  - The *theory of computation*
  - **Computability vs. Complexity**

# Alan Turing (1912-1954)

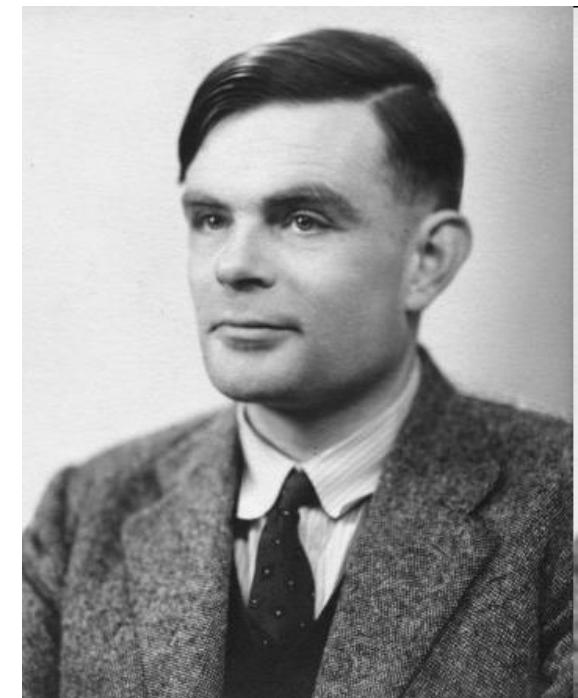
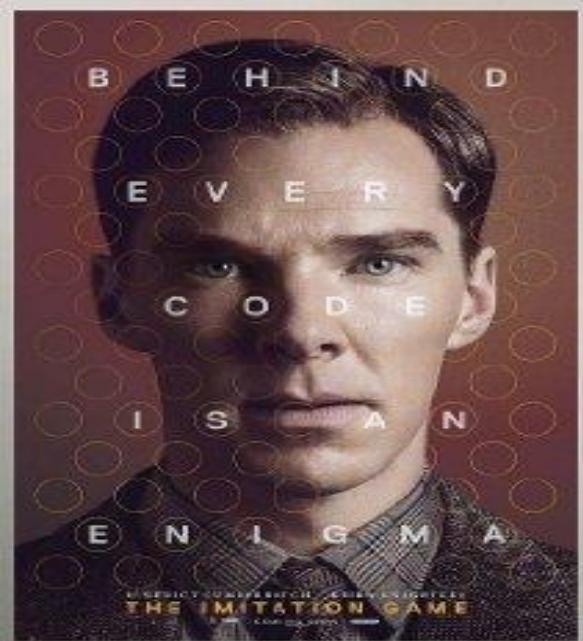
(A pioneer of automata theory)

Father of Modern Computer Science

English mathematician

Studied abstract machines called **Turing machines** even before computers existed

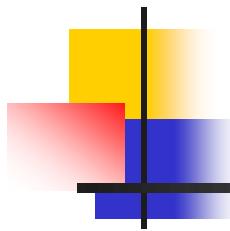
Heard of the Turing test?



# Theory of Computation: A Historical Perspective

1930s	<ul style="list-style-type: none"><li>• Alan Turing studies <b>Turing machines</b></li><li>• <b>Decidability</b></li><li>• <b>Halting problem</b></li></ul>
1940-1950s	<ul style="list-style-type: none"><li>• “<b>Finite automata</b>” machines studied</li><li>• Noam Chomsky proposes the <b>“Chomsky Hierarchy”</b> for formal languages</li></ul>
1969	Cook introduces “intractable” problems or <b>“NP-Hard”</b> problems
1970-	Modern computer science: <b>compilers</b> , <b>computational &amp; complexity theory</b> evolve

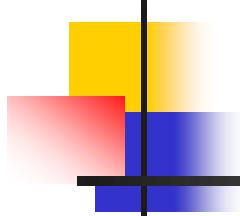
# Basic Building Blocks



# Symbol

---

- a, b, c...A, B, C...0,1,2.....

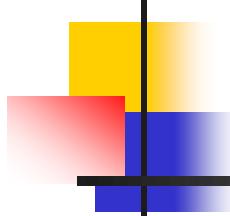


# Alphabet

---

*An alphabet is a finite, non-empty set of symbols*

- We use the symbol  $\Sigma$  (sigma) to denote an alphabet
- Examples:
  - Binary:  $\Sigma = \{0,1\}$
  - All lower case letters:  $\Sigma = \{a,b,c,\dots,z\}$
  - Alphanumeric:  $\Sigma = \{a-z, A-Z, 0-9\}$
  - DNA molecule letters:  $\Sigma = \{a,c,g,t\}$
  - ...



# Strings

---

A *string or word* is a *finite sequence of symbols chosen from  $\Sigma$*

- $\Sigma = \{a, b\}$
- $ab, bab$
- **Empty string is  $\varepsilon$  (or “epsilon”)**
- $aba\varepsilon ba$  or  $ababa$

# Alphabets and Strings

An alphabet is a set of symbols

Example Alphabet:  $\Sigma = \{a, b\}$

A string is a sequence of symbols from the alphabet

Example Strings

$a$

$ab$

$abba$

$aaabbbaaba$   $b$

$u = ab$

$v = bbbaaa$

$w = abba$

Decimal numbers alphabet  $\Sigma = \{0,1,2,\dots,9\}$

102345

567463386

Binary numbers alphabet  $\Sigma = \{0,1\}$

100010001

101101111

Unary numbers alphabet  $\Sigma = \{1\}$

Unary number: 1    11    111    1111    11111

Decimal number: 1    2    3    4    5

# String Operations

 $w = a_1 a_2 \cdots a_n$  $abba$  $v = b_1 b_2 \cdots b_m$  $bbbaaa$ 

## Concatenation

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

$w = a_1 a_2 \cdots a_n$  $ababa aabb b$ 

Reverse

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

# String Length

$$w = a_1 a_2 \cdots a_n$$

Length:  $|w| = n$

Examples:  $|abba| = 4$

$$|aa| = 2$$

$$|a| = 1$$

# Empty String

A string with no letters is denoted:  $\lambda$  or  $\epsilon$

Observations:  $|\lambda| = 0$

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

$$\lambda abba = abba \lambda = ab\lambda ba = abba$$

# Substring

Substring of string:  
a subsequence of consecutive characters

String

abbab

abbab

abbab

abbab

Substring

*ab*

*abba*

*b*

*bbab*

# Prefix and Suffix

*abbab*

Prefixes

$\lambda$

$a$

$ab$

$abb$

$abba$

$abbab$

Suffixes

*abbab*

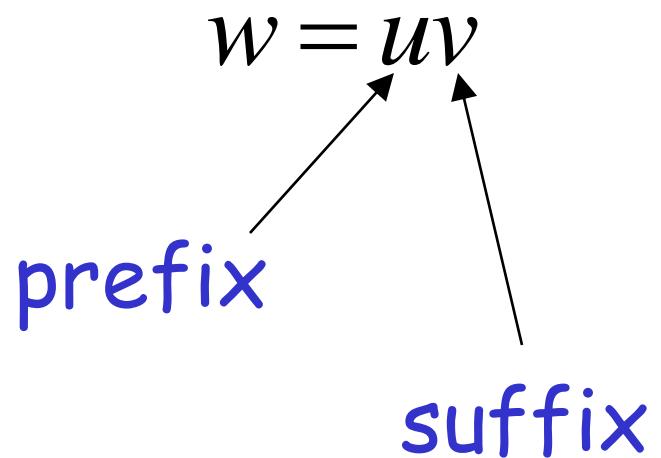
*bbab*

*bab*

*ab*

*b*

$\lambda$



# Another Operation

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

Example:  $(abba)^2 = abbaabba$

Definition:  $w^0 = \lambda$

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

# The \* Operation

$\Sigma^*$ : the set of all possible strings from alphabet  $\Sigma$

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

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

# The + Operation

$\Sigma^+$  : the set of all possible strings from alphabet  $\Sigma$  except  $\lambda$

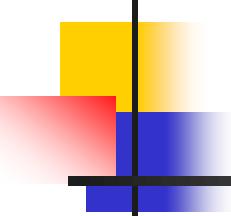
$$\Sigma = \{a, b\}$$

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

$$\Sigma^+ = \Sigma^* - \lambda$$

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

# Languages



# Languages

---

*L is said to be a language over alphabet  $\Sigma$ , only if  $L \subseteq \Sigma^*$*

→ this is because  $\Sigma^*$  is the set of all strings (of all possible length including 0) over the given alphabet  $\Sigma$

Examples:       $\Sigma=\{a, b\}$

- L1=set of all string of length 2  
    = $\{aa, ab, ba, bb\}$
- L2=set of all string of length 3  
    = $\{aaa, aab, aba, abb, baa, bab, bba, bbb\}$
- L3=set of all string where each string starts with 'a'  
    = $\{a, aa, ab, aaa, aab, aba, abb, \dots\}$

**Language:** a set of strings

In mathematics, computer science, and linguistics, a formal language consists of words whose letters are taken from an alphabet and are well-formed according to a specific set of rules.

**String:** a sequence of symbols

from some alphabet

Example: Alphabet:  $\Sigma = \{a, b, c, \dots, z\}$

Strings: cat, dog, house

Language: {cat, dog, house}

A language over alphabet  $\Sigma$   
is any subset of  $\Sigma^*$

Examples:

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

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

Language:  $\{\lambda\}$

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

Language:  $\{\lambda, abba, baba, aa, ab, aaaaaaa\}$

Languages are used to describe computation problems:

$$PRIMES = \{2, 3, 5, 7, 11, 13, 17, \dots\}$$

$$EVEN = \{0, 2, 4, 6, \dots\}$$

Alphabet:  $\Sigma = \{0, 1, 2, \dots, 9\}$

# More Language Examples

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

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

$\lambda$

$ab$

$aabb$

$aaaaabbbbb$

$\} \in L$

$abb \notin L$

# Prime numbers

Alphabet  $\Sigma = \{0,1,2,\dots,9\}$

Language:

$PRIMES = \{x : x \in \Sigma^* \text{ and } x \text{ is prime}\}$

$PRIMES = \{2,3,5,7,11,13,17,\dots\}$

# Even and odd numbers

Alphabet  $\Sigma = \{0,1,2,\dots,9\}$

$EVEN = \{x : x \in \Sigma^* \text{ and } x \text{ is even}\}$

$EVEN = \{0,2,4,6,\dots\}$

$ODD = \{x : x \in \Sigma^* \text{ and } x \text{ is odd}\}$

$ODD = \{1,3,5,7,\dots\}$

# Unary Addition

Alphabet:  $\Sigma = \{1, +, =\}$

Language:

$ADDITION = \{x + y = z : x = 1^n, y = 1^m, z = 1^k,$   
 $n + m = k\}$

$11 + 111 = 11111 \in ADDITION$

$111 + 111 = 111 \notin ADDITION$

Note that:

Sets

$$\emptyset = \{ \ } \neq \{\lambda\}$$

Set size

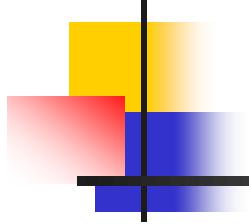
$$|\{ \ }| = |\emptyset| = 0$$

Set size

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

String length

$$|\lambda| = 0$$



# Language

---

## Examples:

1. Let L be *the language of all strings consisting of n 0's followed by n 1's*:

$$L = \{\varepsilon, 01, 0011, 000111, \dots\}$$

2. Let L be *the language of all strings of with equal number of 0's and 1's*:

$$L = \{\varepsilon, 01, 10, 0011, 1100, 0101, 1010, 1001, \dots\}$$

**Definition:**  $\emptyset$  denotes the Empty language

- Let  $L = \{\varepsilon\}$ ; Is  $L = \emptyset$ ?

# Powers of an alphabet

Let  $\Sigma$  be an alphabet.

- $\Sigma^k$  = the set of all strings of length  $k$
- $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \dots$
- $\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \dots$
- $\Sigma = \{a, b\}$
- $\Sigma^1 = \{a, b\}$
- $\Sigma^2 = \{aa, ab, ba, bb\}$
- $\Sigma^3 = \{aaa, aab, aba, baa, bbb, bba, bab, abb\}$
- $\Sigma^* = \{\epsilon\} \cup \{a, b\} \cup \{aa, ab, ba, bb\} \cup \dots$
- $\Sigma^* = \{\epsilon, a, b, aa, ab, ba, bb, aa, aab, aba, baa, bbb, bba, bab, \dots\}$

- Language is finite

$\Sigma = \{a, b\}$   $L_1$ =set of all string of length 2

$$L_1 = \{aa, ab, ba, bb\}$$

- Language is infinite

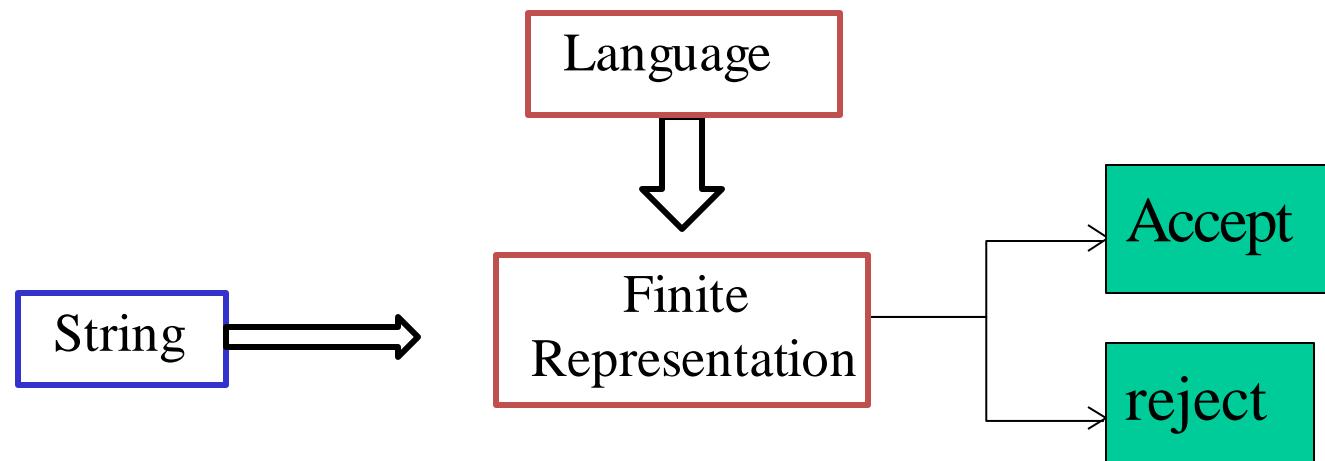
$\Sigma = \{a, b\}$   $L_3$ =set of all string where each string starts with ‘a

$$L_2 = \{a, aa, ab, abb, aab, aba, \dots\}$$

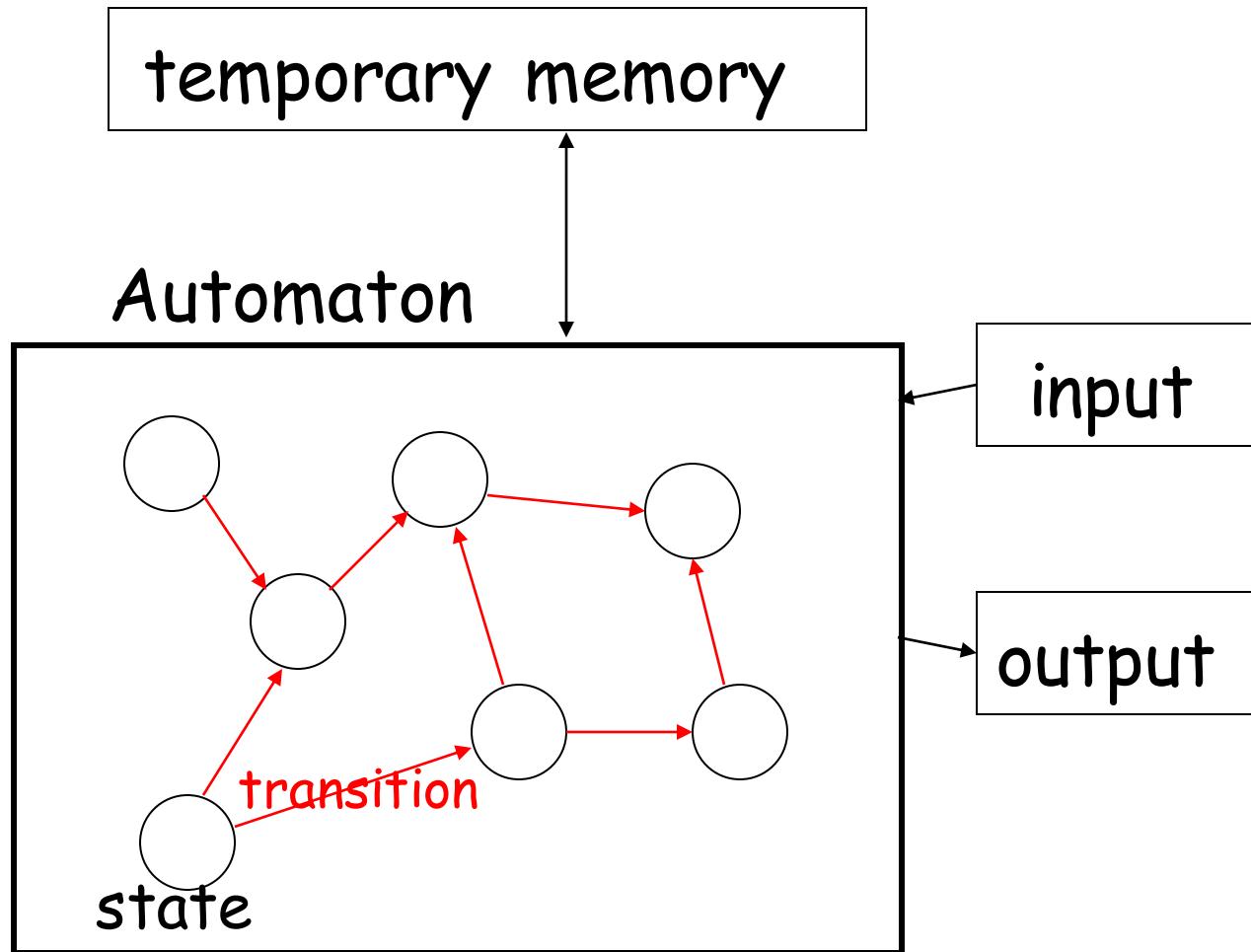
- String

- Linear searching have limitations so.....

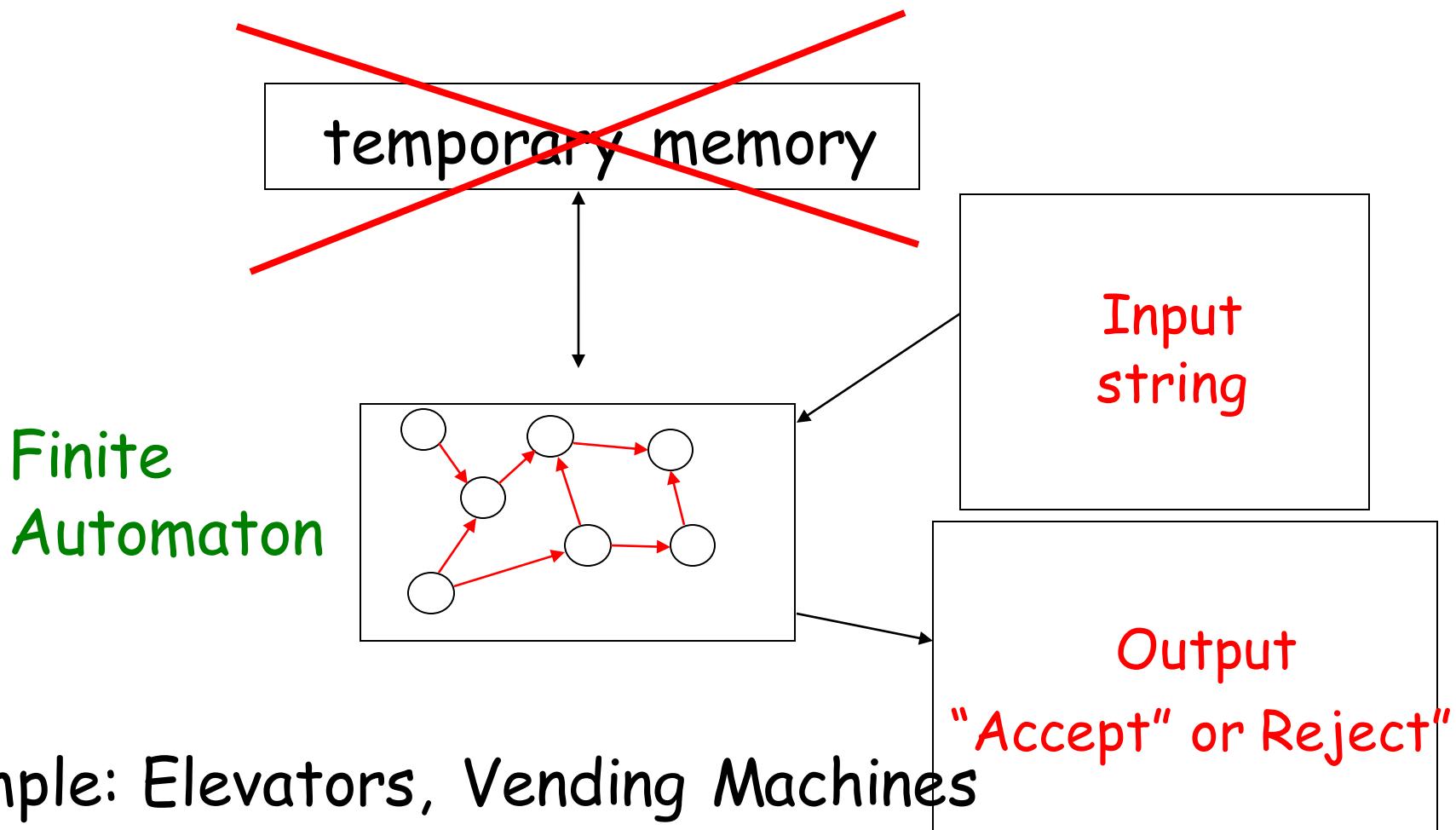
•.



# Automaton



# Finite Automaton/Finite State Machine

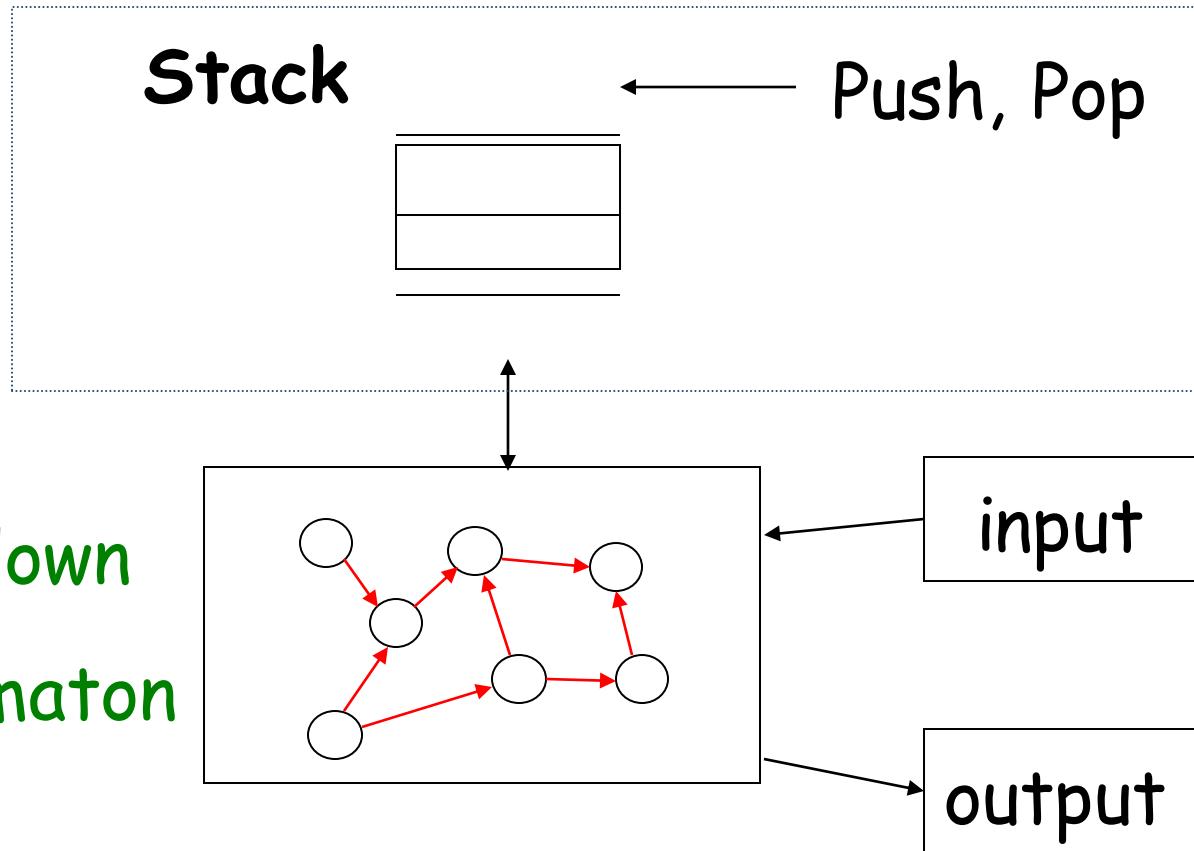


Example: Elevators, Vending Machines  
(small computing power)

# Pushdown Automaton

Temp.  
memory

Pushdown  
Automaton

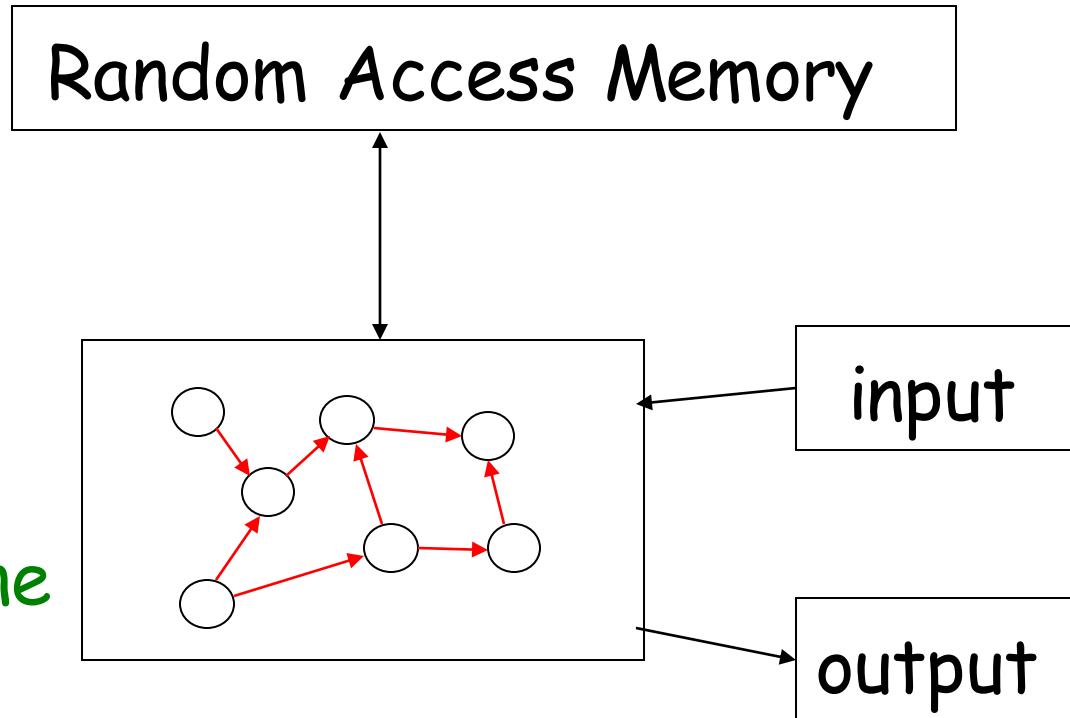


Example: Compilers for Programming Languages  
(medium computing power)

# Turing Machine

Temp.  
memory

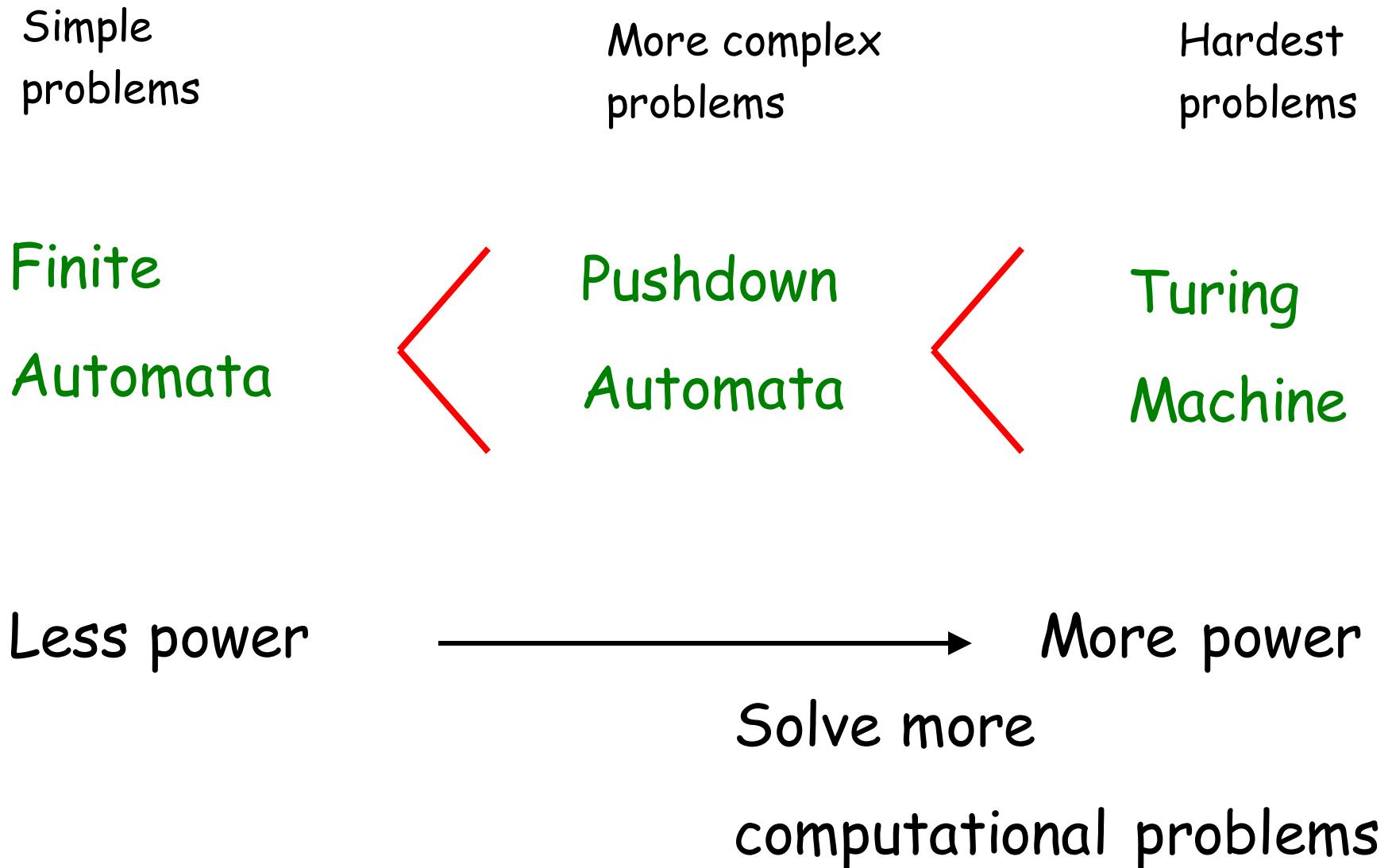
Turing  
Machine



Examples: Computer

(highest computing power)

# Power of Automata



## Finite Automata

Finite Automata with o/p

Moore  
Machine

Mealy  
Machine

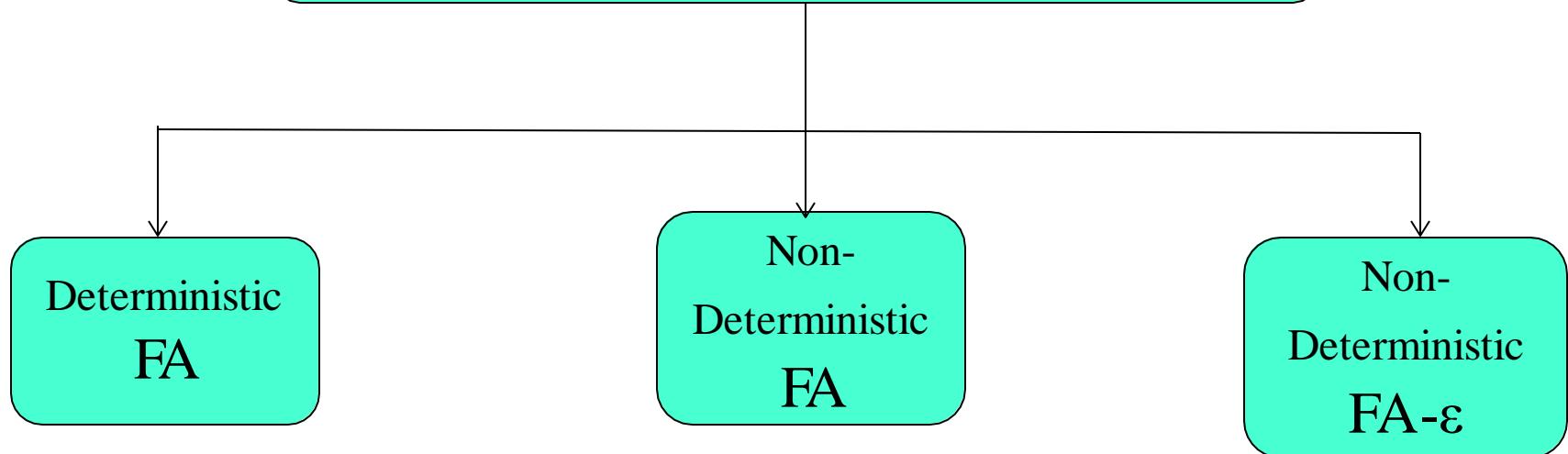
Finite Automata without o/p

Deterministic  
FA

Non Deterministic  
FA- $\epsilon$

Non-  
Deterministic  
FA

## Finite Automata without o/p



# Formal Definition

## Deterministic Finite Automaton (DFA)

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

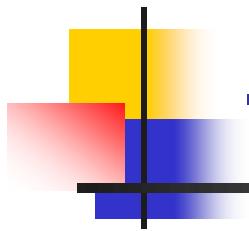
$Q$  : set of states

$\Sigma$  : input alphabet     $\lambda \notin \Sigma$

$\delta$  : transition function

$q_0$  : initial state

$F$  : set of accepting states



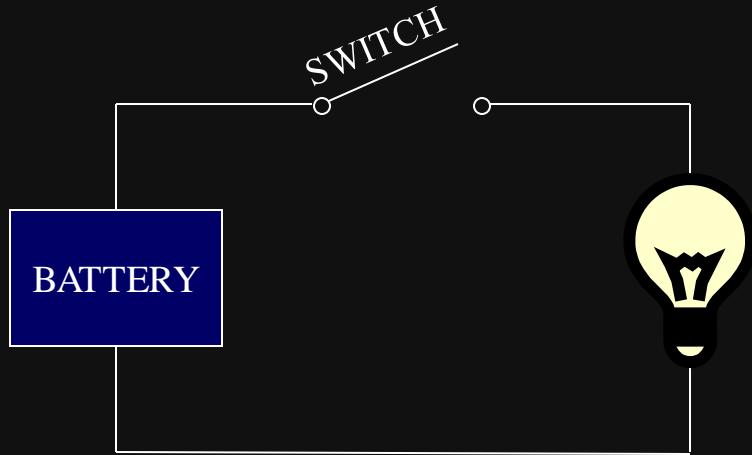
# Deterministic Finite Automata

## - Definition

- A Deterministic Finite Automaton (DFA) consists of:
  - $Q \implies$  a finite set of states
  - $\Sigma \implies$  a finite set of input symbols (alphabet)
  - $q_0 \implies$  a start state
  - $F \implies$  set of accepting states
  - $\delta \implies$  a transition function, which is a mapping between  $Q \times \Sigma \implies Q$
- A DFA is defined by the 5-tuple:
  - $\{Q, \Sigma, q_0, F, \delta\}$

# A simple computer

---



**input:** switch

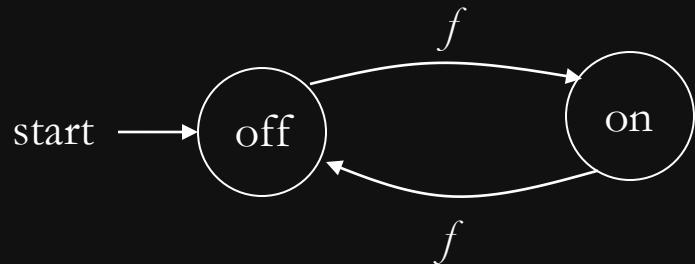
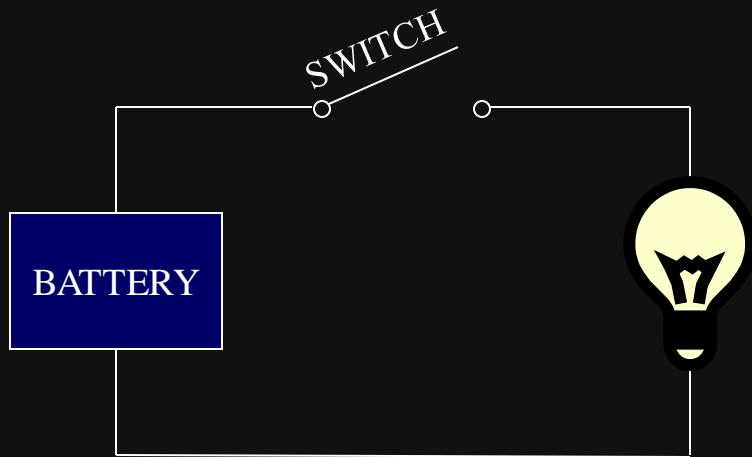
**output:** light bulb

**actions:** flip switch

**states:** on, off

# A simple “computer”

---



**input:** switch

**output:** light bulb

**actions:**  $f$  for “flip switch”

**states:** on, off

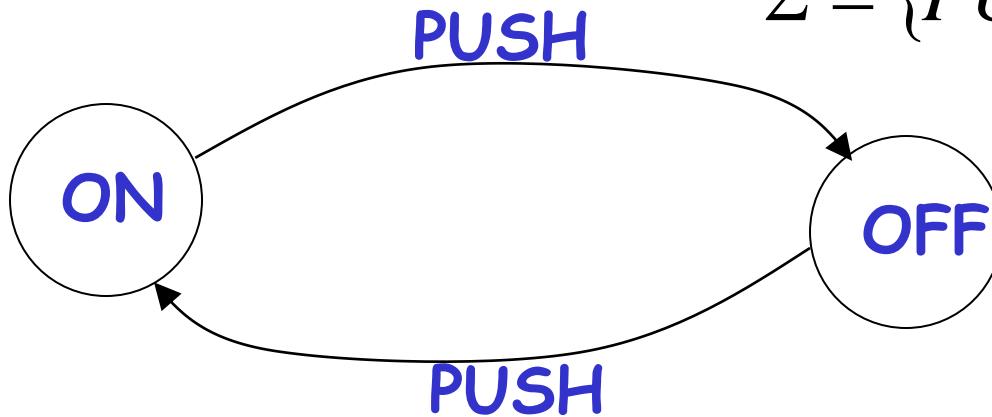
bulb is on if and only if  
there was an **odd** number  
of flips

# State transition systems

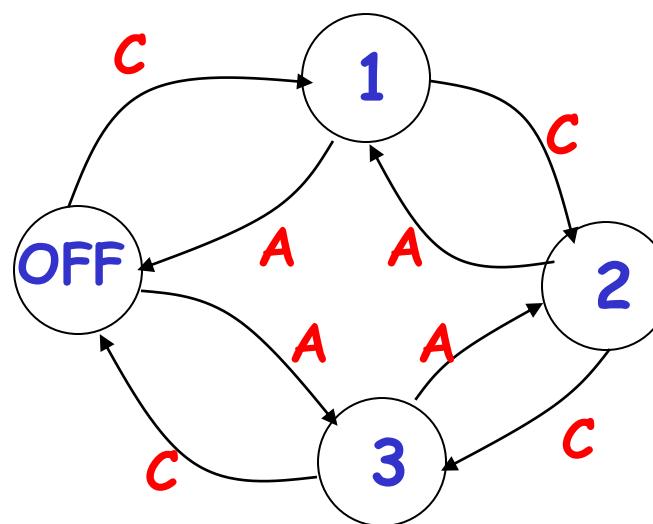
- This is the oldest model, coming from the natural sciences.
  - A system is one of several possible states.
  - Every input or stimulus causes a change of state.
  - Desirable behavior of the system corresponds to partitioning states into two: good and bad.
  - It is reasonable to start the system from a designated state.

•Electric  
Switch

$$Q = \{ON, OFF\}$$
$$\Sigma = \{PUSH\}$$



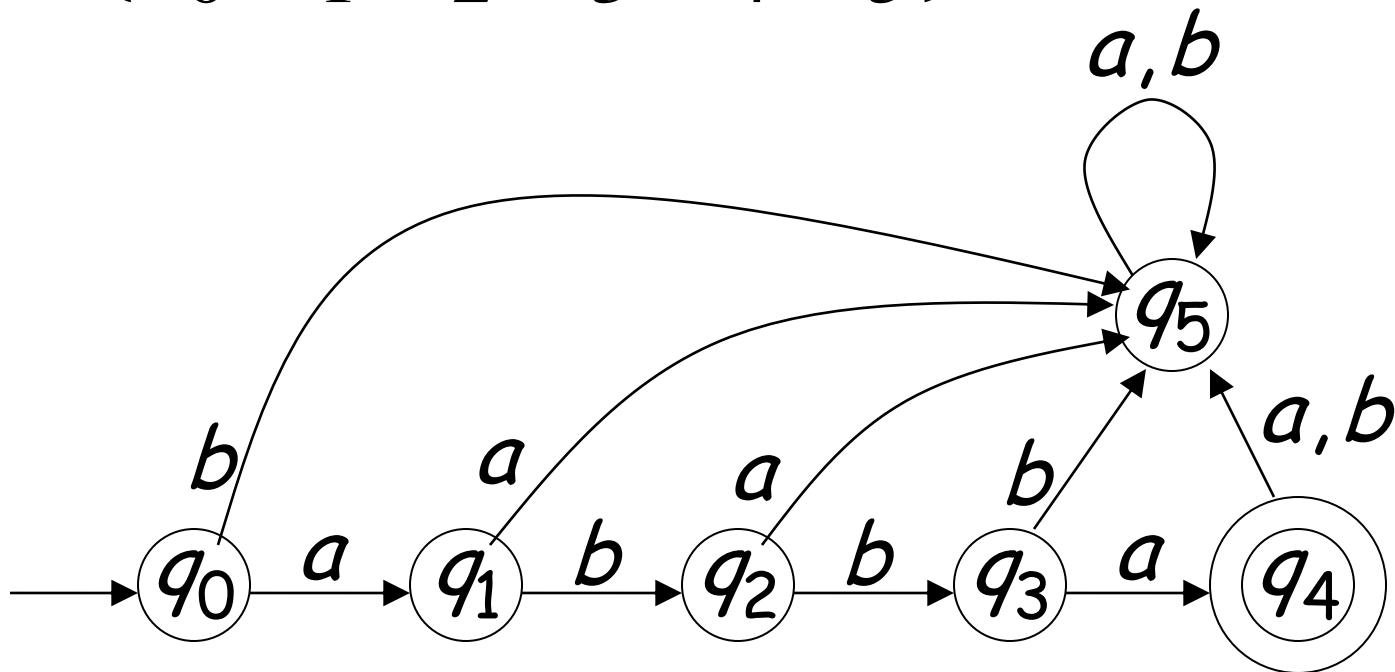
•FAN  
REGULATOR



# Set of States $\mathcal{Q}$

## Example

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

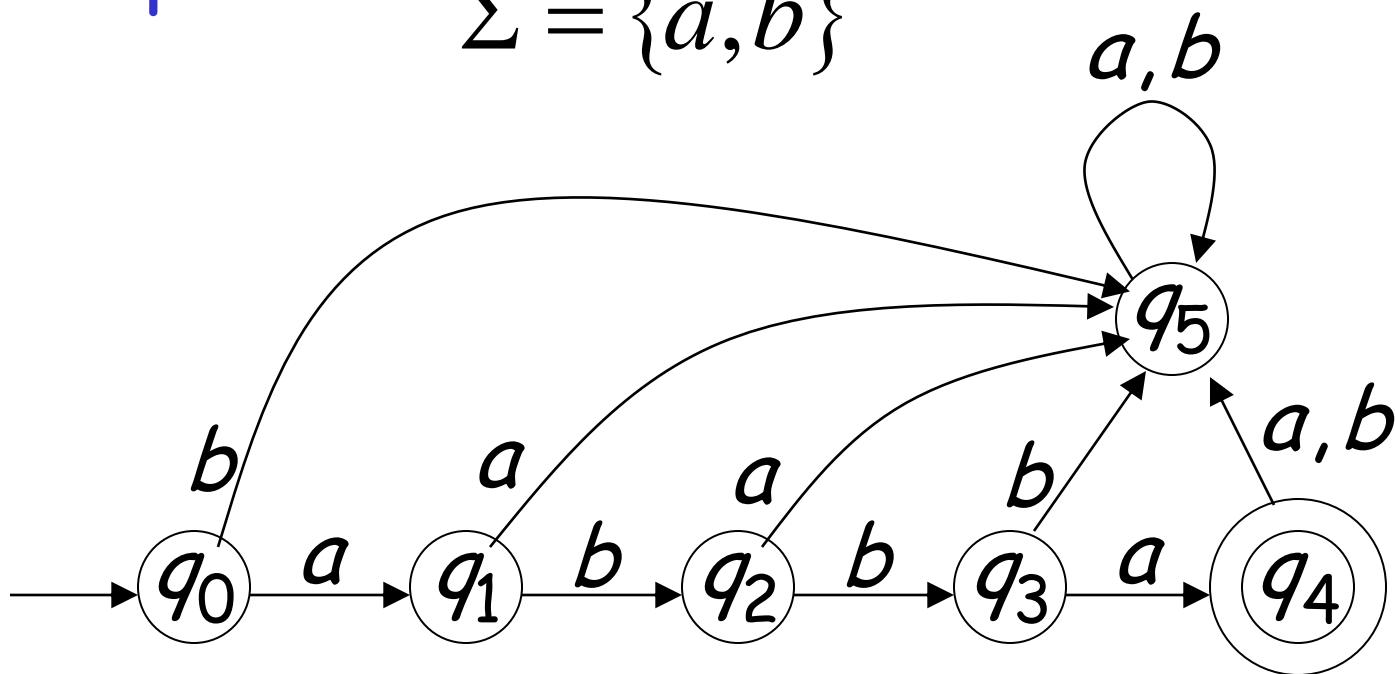


# Input Alphabet $\Sigma$

$\lambda \notin \Sigma$  :the input alphabet never contains  $\lambda$

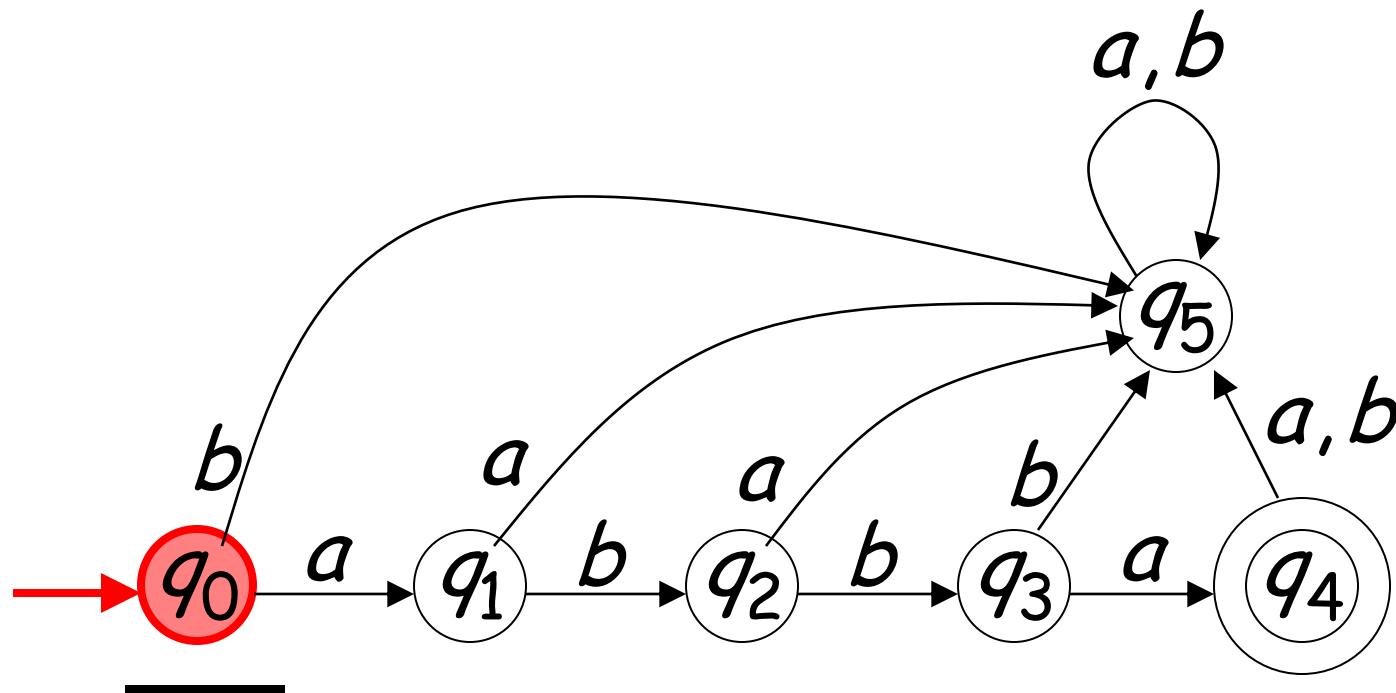
Example

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



# Initial State $q_0$

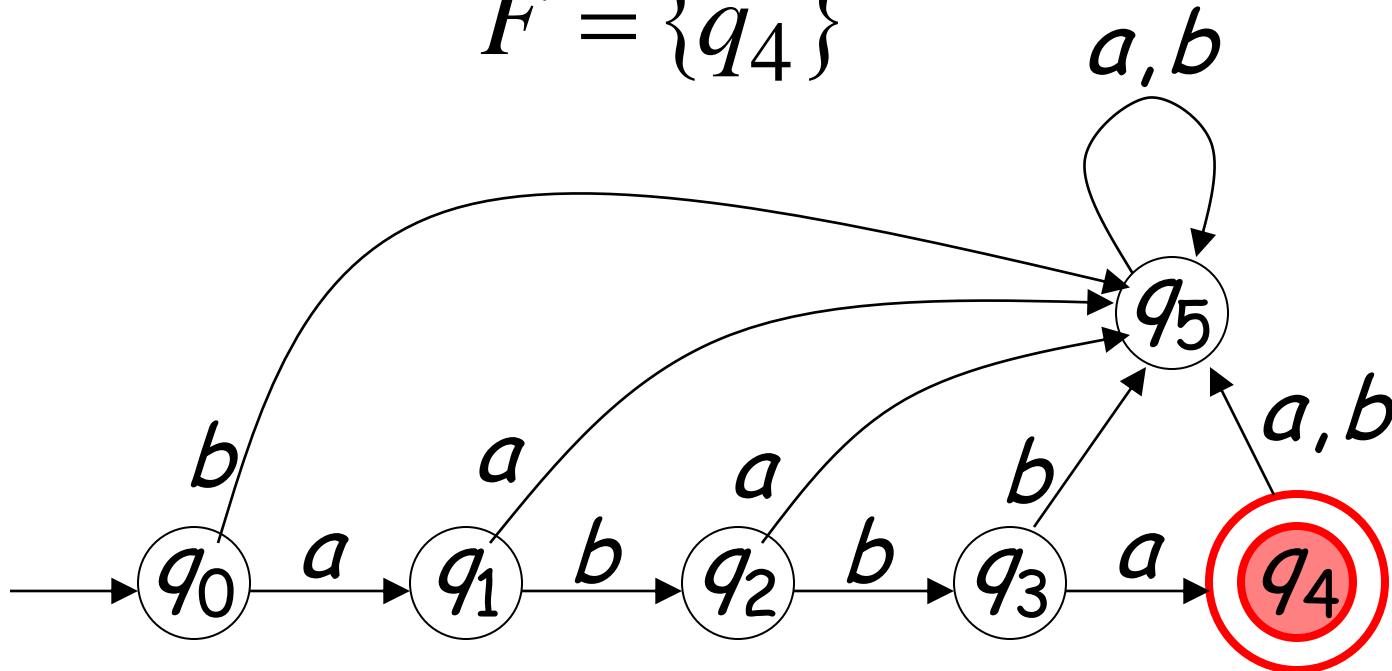
Example



# Set of Accepting States $F \subseteq Q$

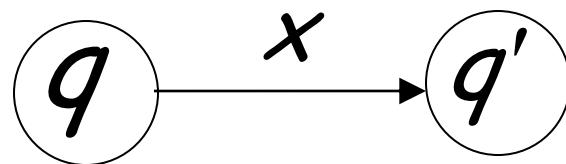
Example

$$F = \{q_4\}$$



Transition Function  $\delta: Q \times \Sigma \rightarrow Q$

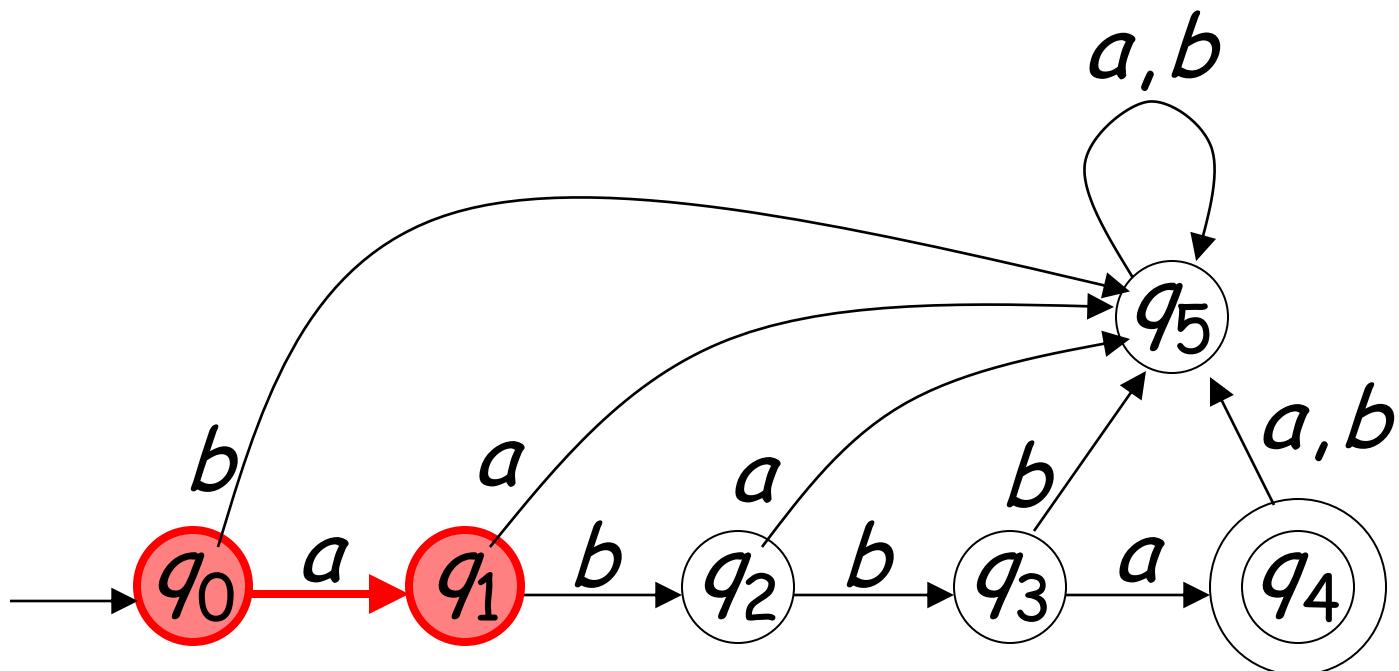
$$\delta(q, x) = q'$$



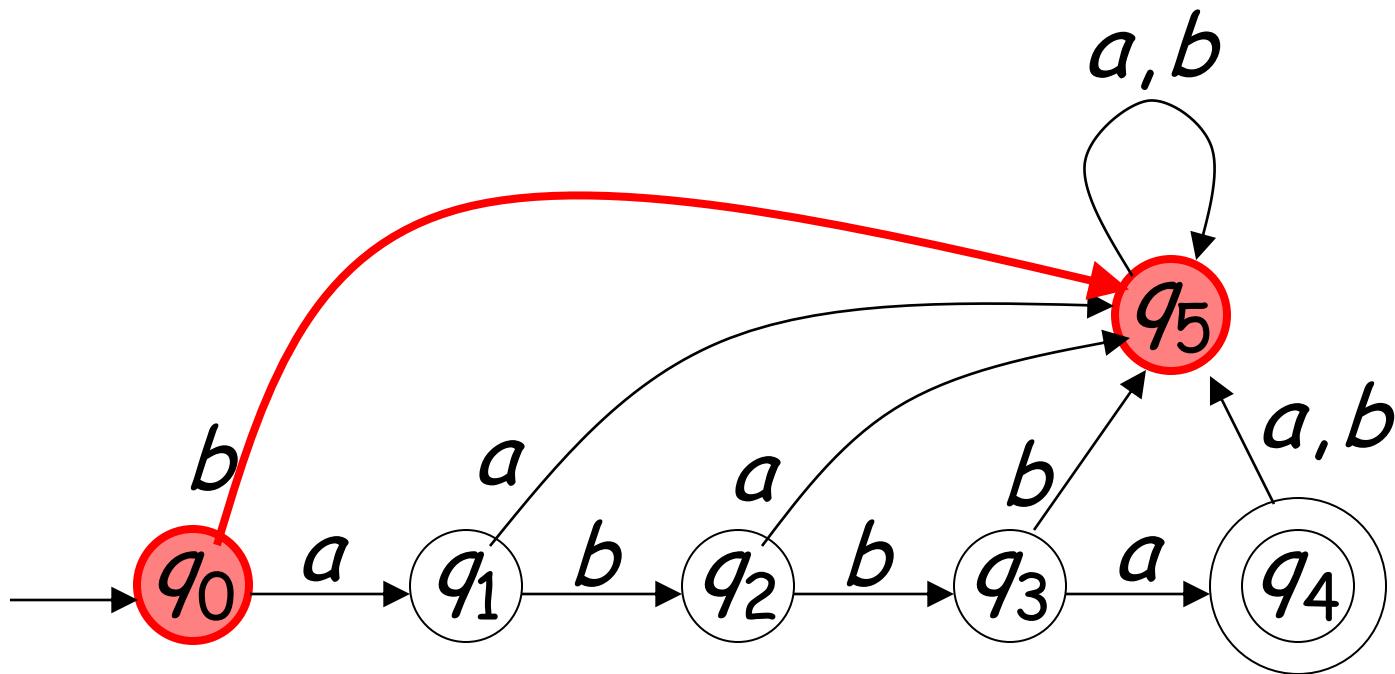
Describes the result of a transition  
from state  $q$  with symbol  $x$

Example:

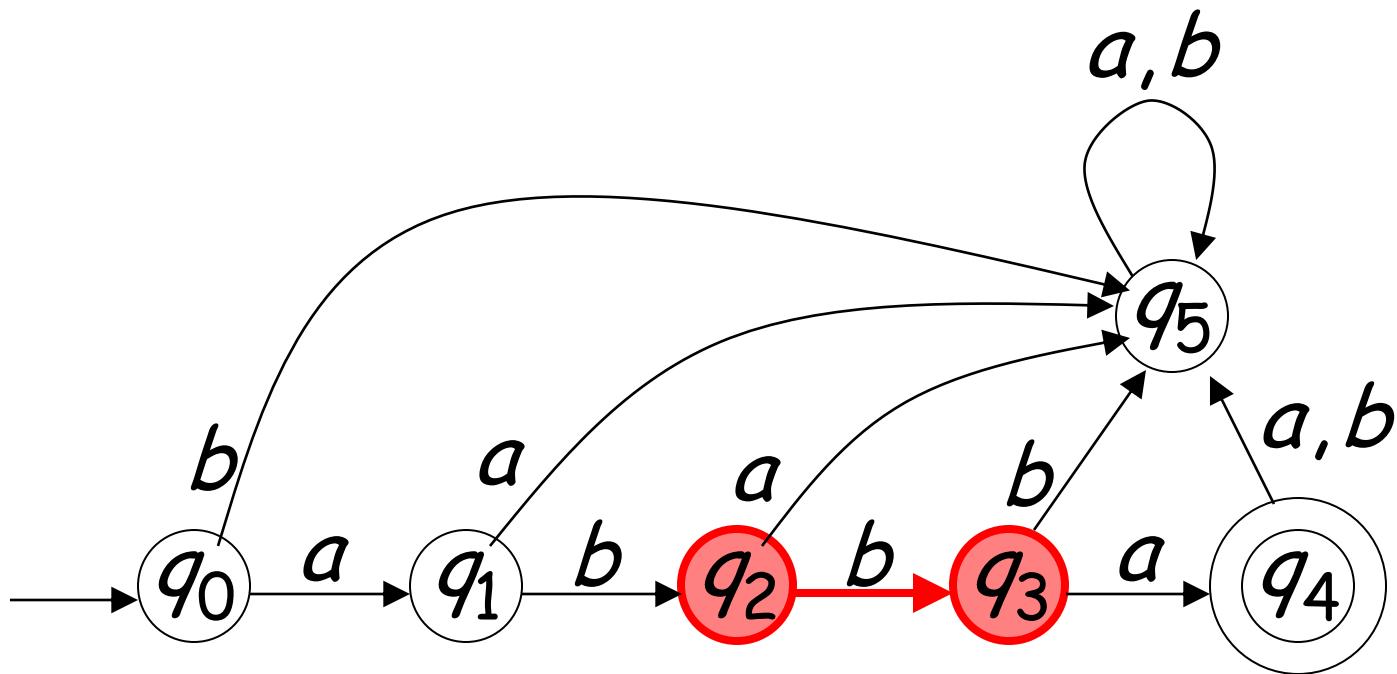
$$\delta(q_0, a) = q_1$$



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



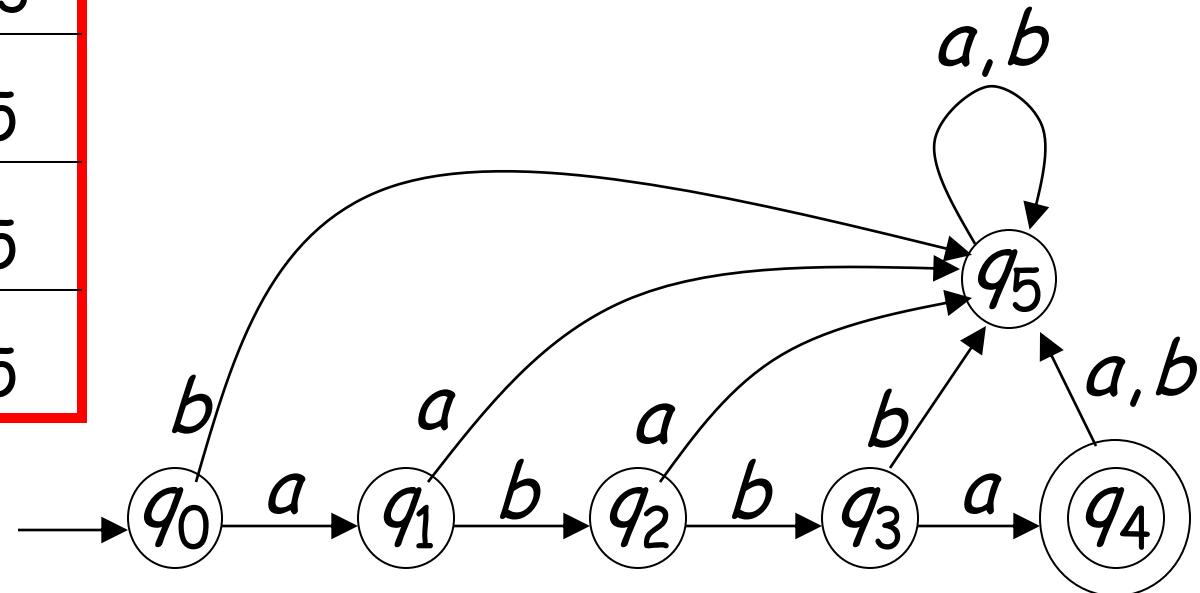
$$\delta(q_2, b) = q_3$$



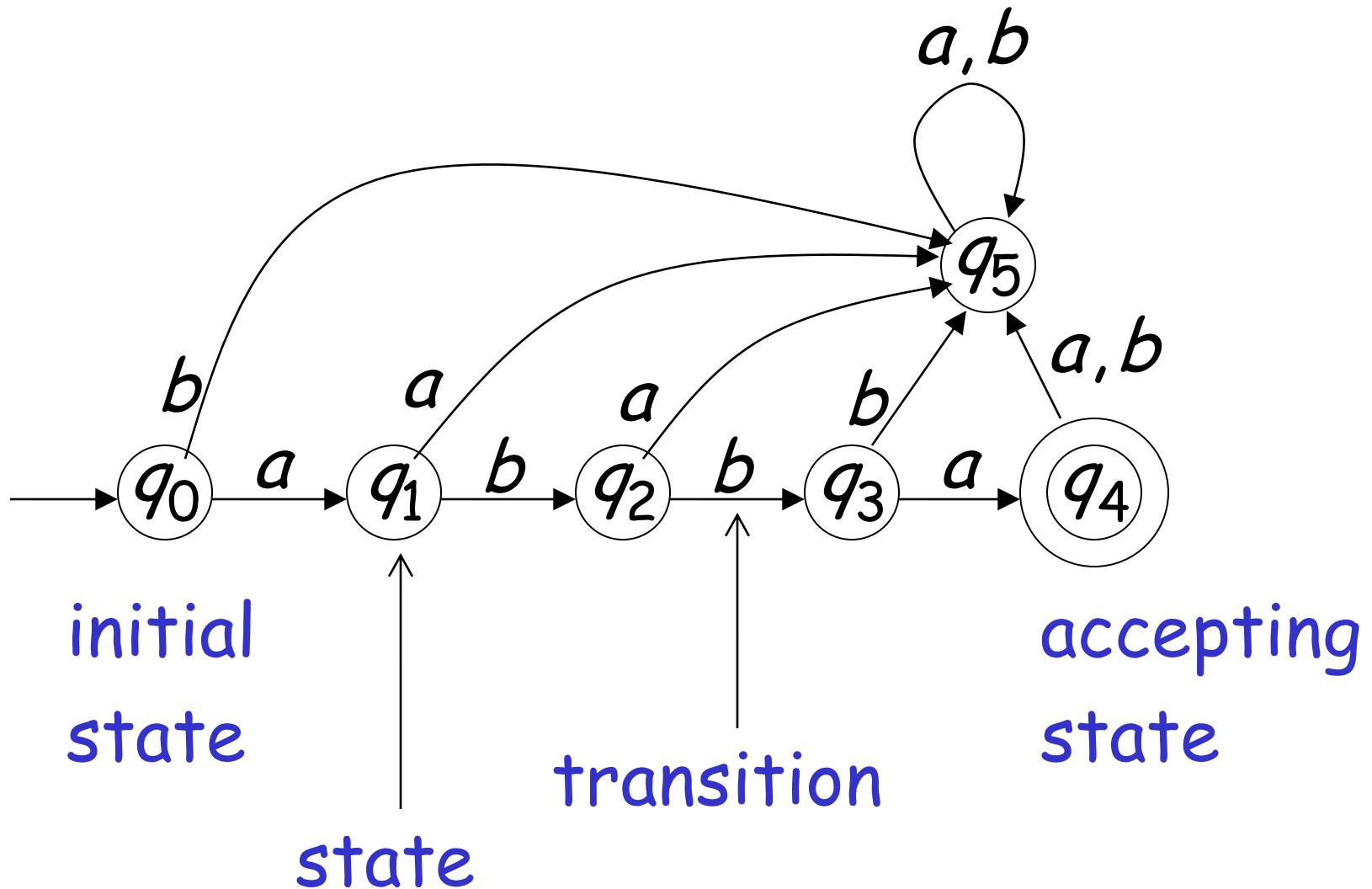
# Transition Table for $\delta$

symbols

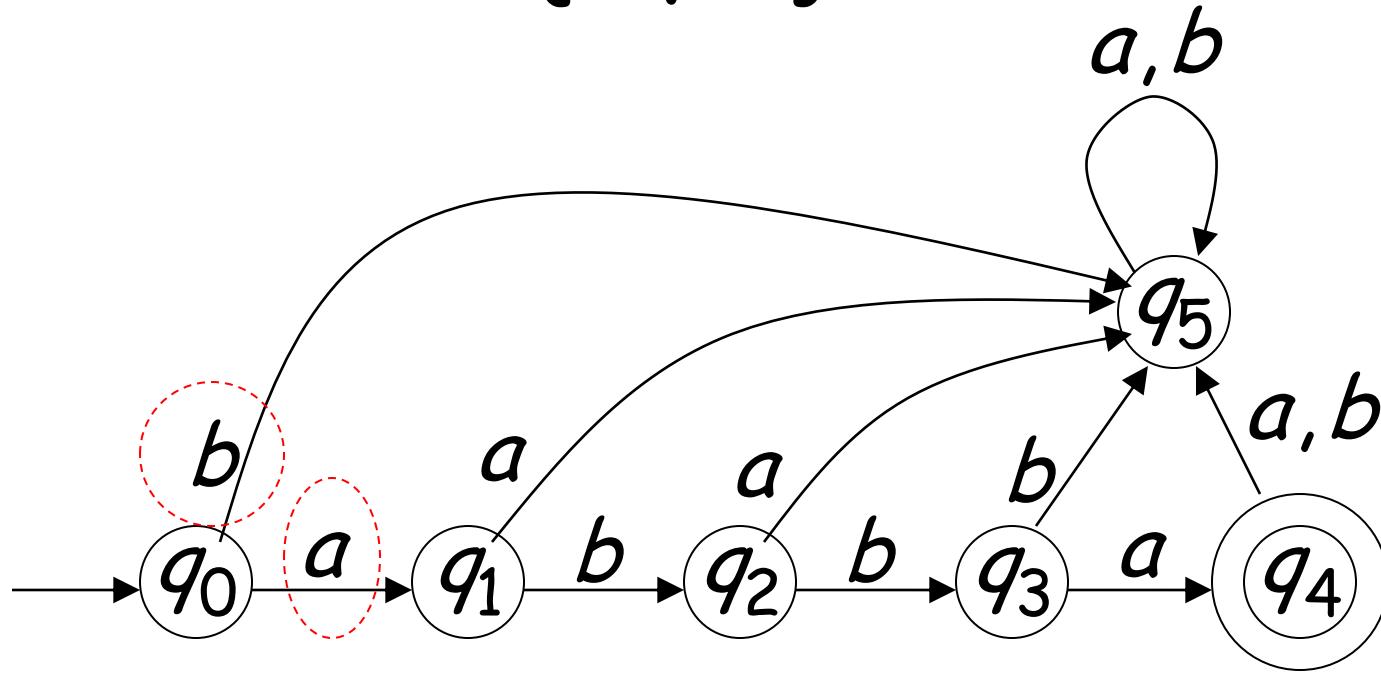
$\delta$	$a$	$b$
$q_0$	$q_1$	$q_5$
$q_1$	$q_5$	$q_2$
$q_2$	$q_5$	$q_3$
$q_3$	$q_4$	$q_5$
$q_4$	$q_5$	$q_5$
$q_5$	$q_5$	$q_5$



# Transition Graph



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



For every state, there is a transition  
for every symbol in the alphabet

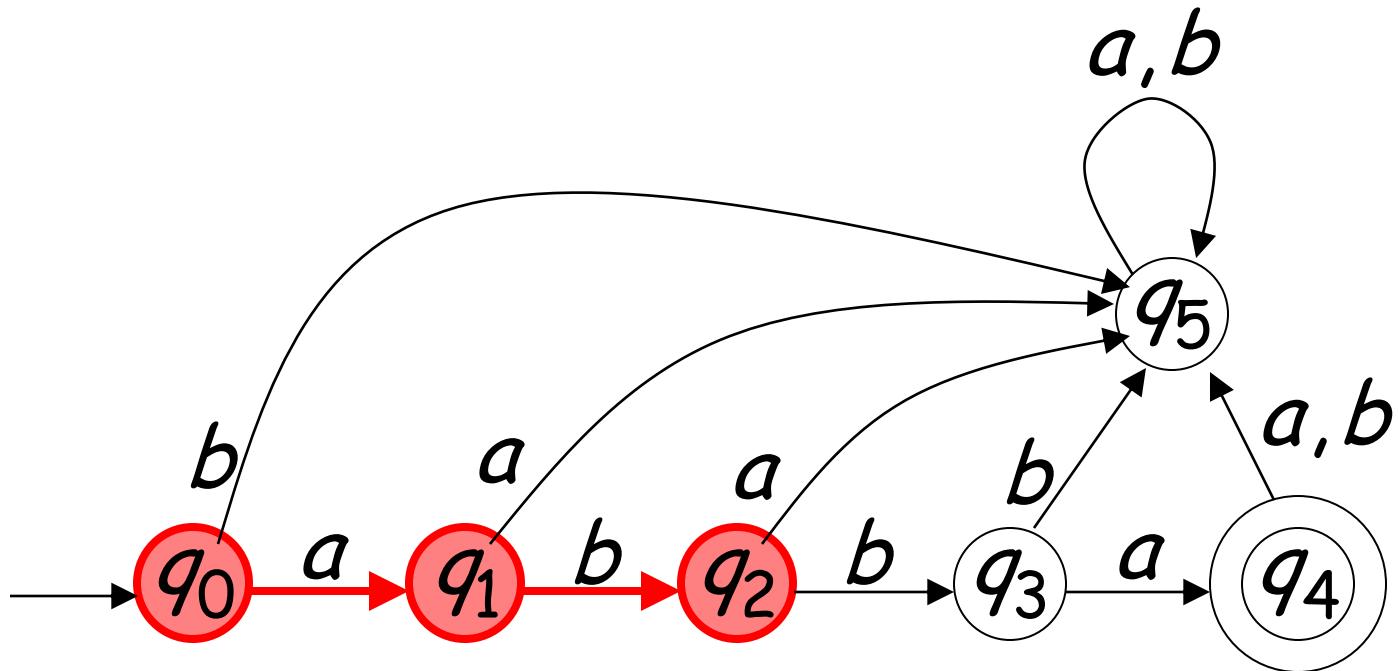
# Extended Transition Function

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

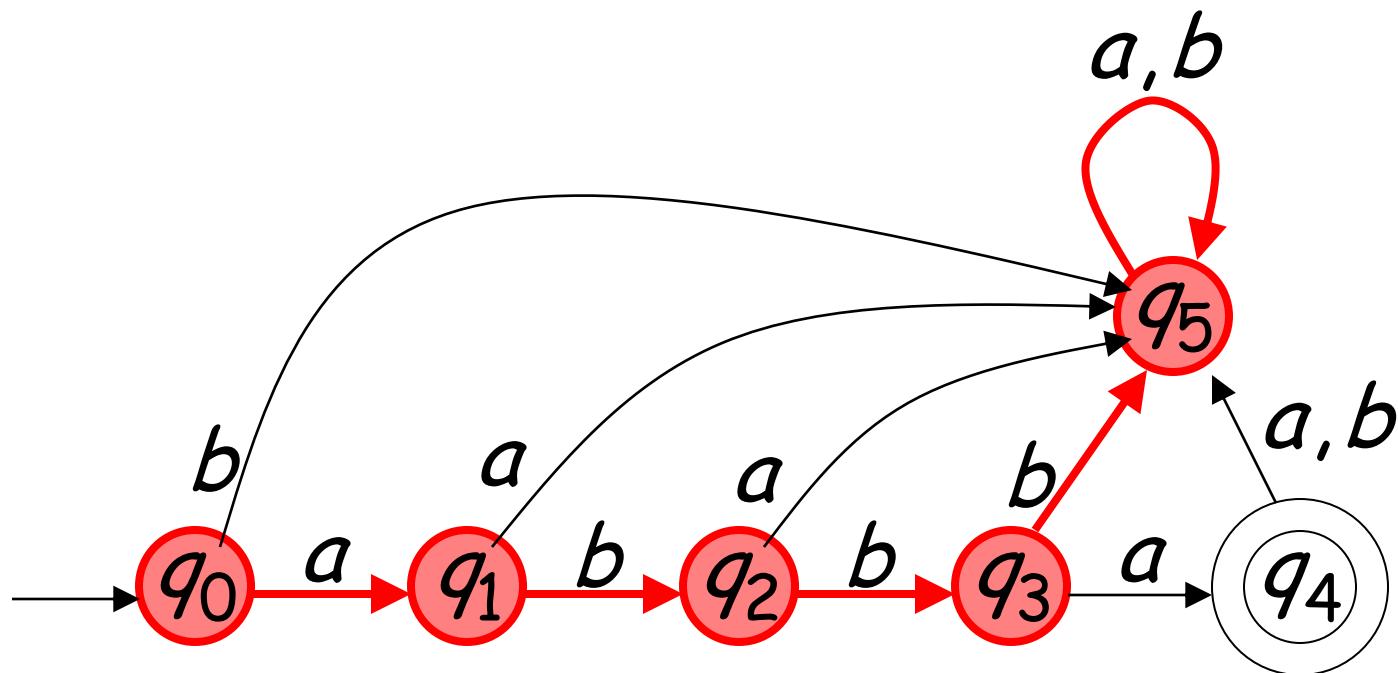
$$\delta^*(q, w) = q'$$

Describes the resulting state  
after scanning string  $w$  from state  $q$

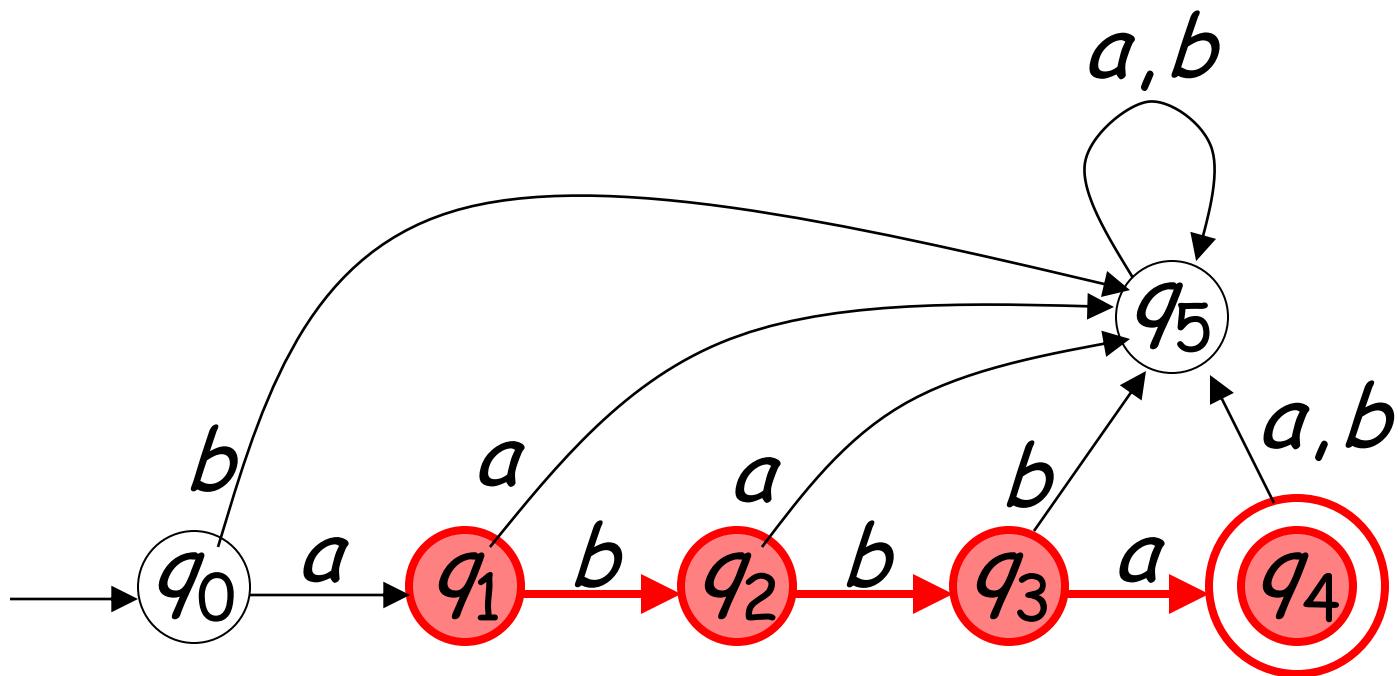
Example:  $\delta^*(q_0, ab) = q_2$



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



$$\delta^*(q_1, bba) = q_4$$



Special case:

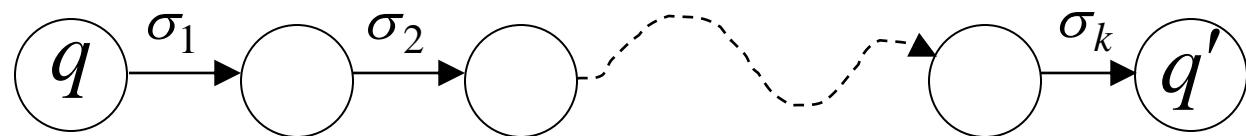
for any state  $q$

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

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

implies that there is a walk of transitions

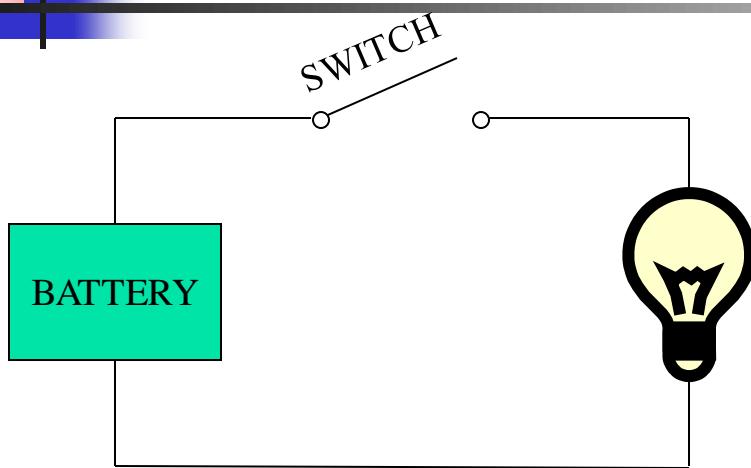
$$w = \sigma_1 \sigma_2 \cdots \sigma_k$$



states may be repeated



# A simple “Finite State Machine”

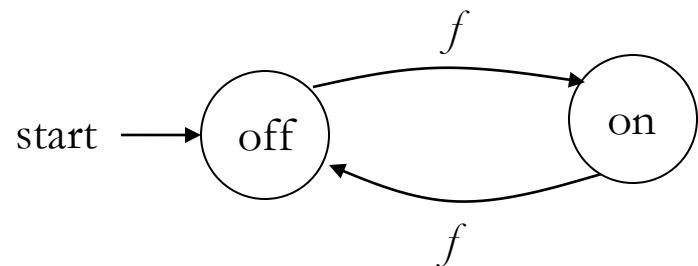


**input:** switch

**output:** light bulb

**actions:**  $f$  for “flip switch”

**states:** on, off



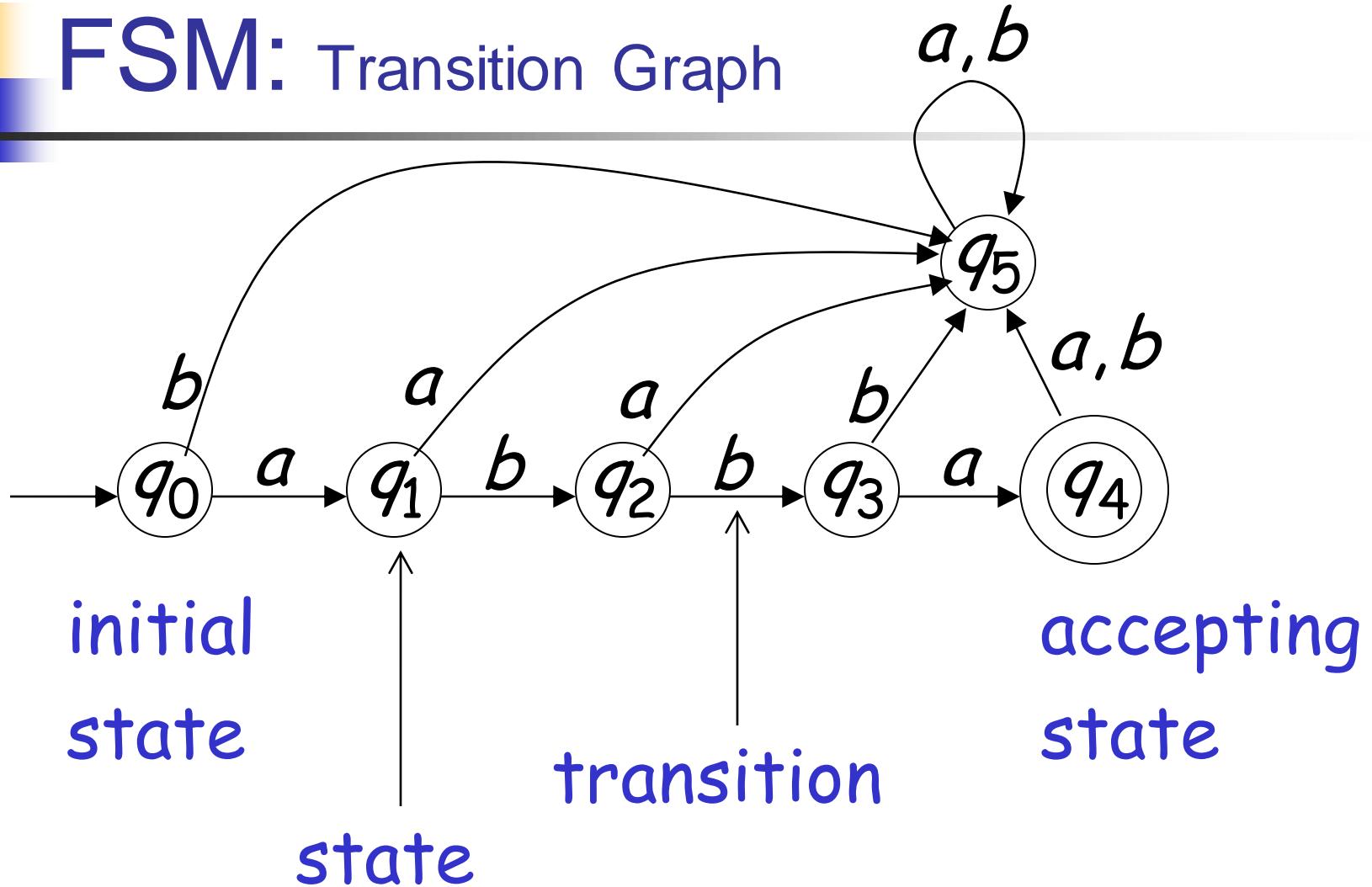
bulb is on if and only if  
there was an odd number  
of flips

$$\{Q, \Sigma, q_0, A, \delta, \}$$

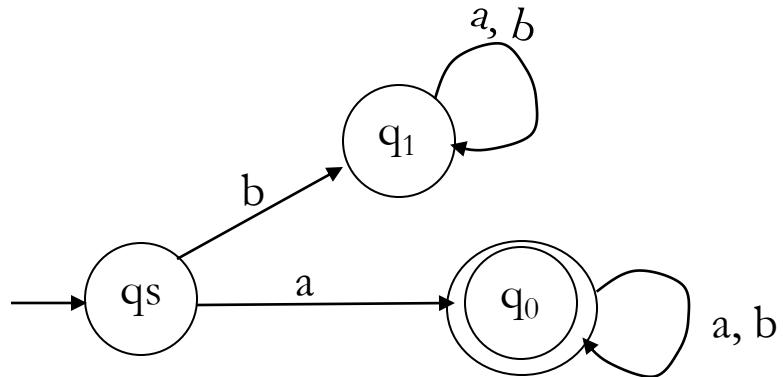
$$\{\{on, off\}, \{f\}, off, on, \delta, \}$$

$$\delta(off, f) = on, \delta(on, f) = off$$

# FSM: Transition Graph



- Examples: Construct a DFA that accepts the language
- $L = \{\text{Set of all strings started with symbol 'a' over the alphabet } \{a,b\}\}$
- $\Sigma = \{a, b\}$
- Answer:-  $L=\{a, ab, aa, aab, aba, aaa, abb, \dots\}$



# Examples

---

- Construct a DFA that accepts the language

$$L = \{010, 1\} \quad (\Sigma = \{0, 1\})$$

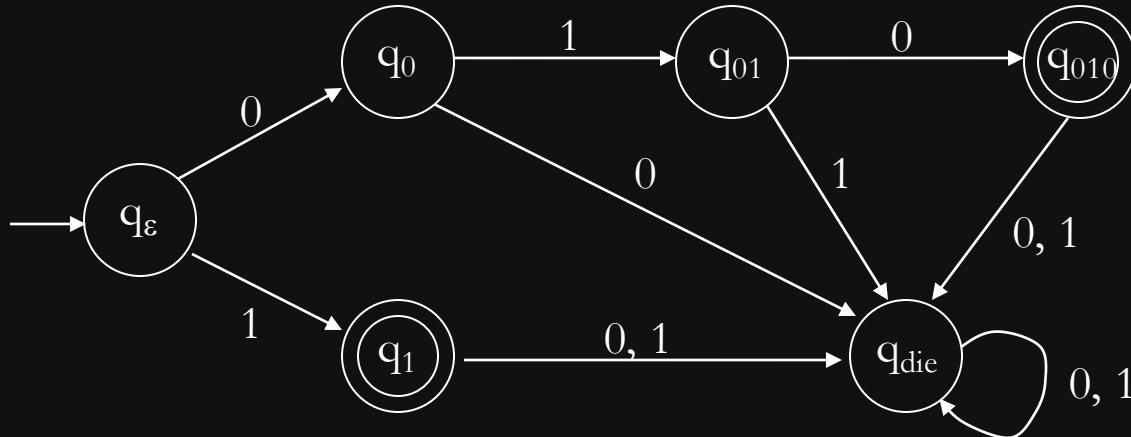
# Examples

---

- Construct a DFA that accepts the language

$$L = \{010, 1\} \quad (\Sigma = \{0, 1\})$$

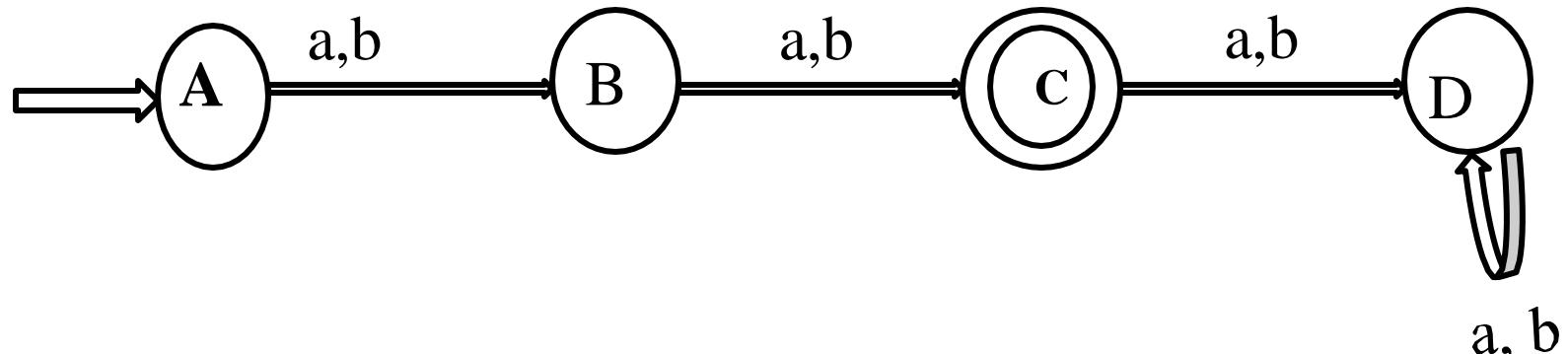
- Answer



## Example

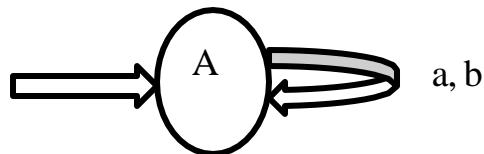
Construct a DFA which accepts set of all string over  $\Sigma = \{a, b\}$  of length 2.

$$L = \{aa, ab, ba, bb\}$$



String Accept:- scan the entire string, if we reach a final state from initial state.  
language Accept:- All the string in language are "accepted" and all string which are not in the language are "rejected"

## Example

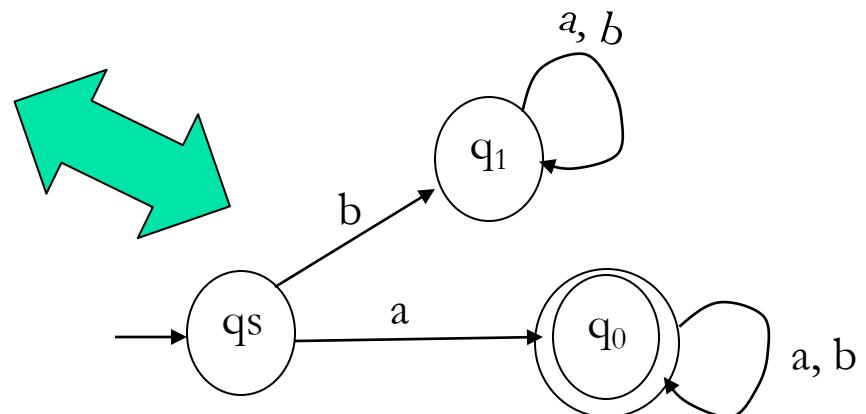


Above DFA is incorrect because it also accept the string which is not in language



# Summary

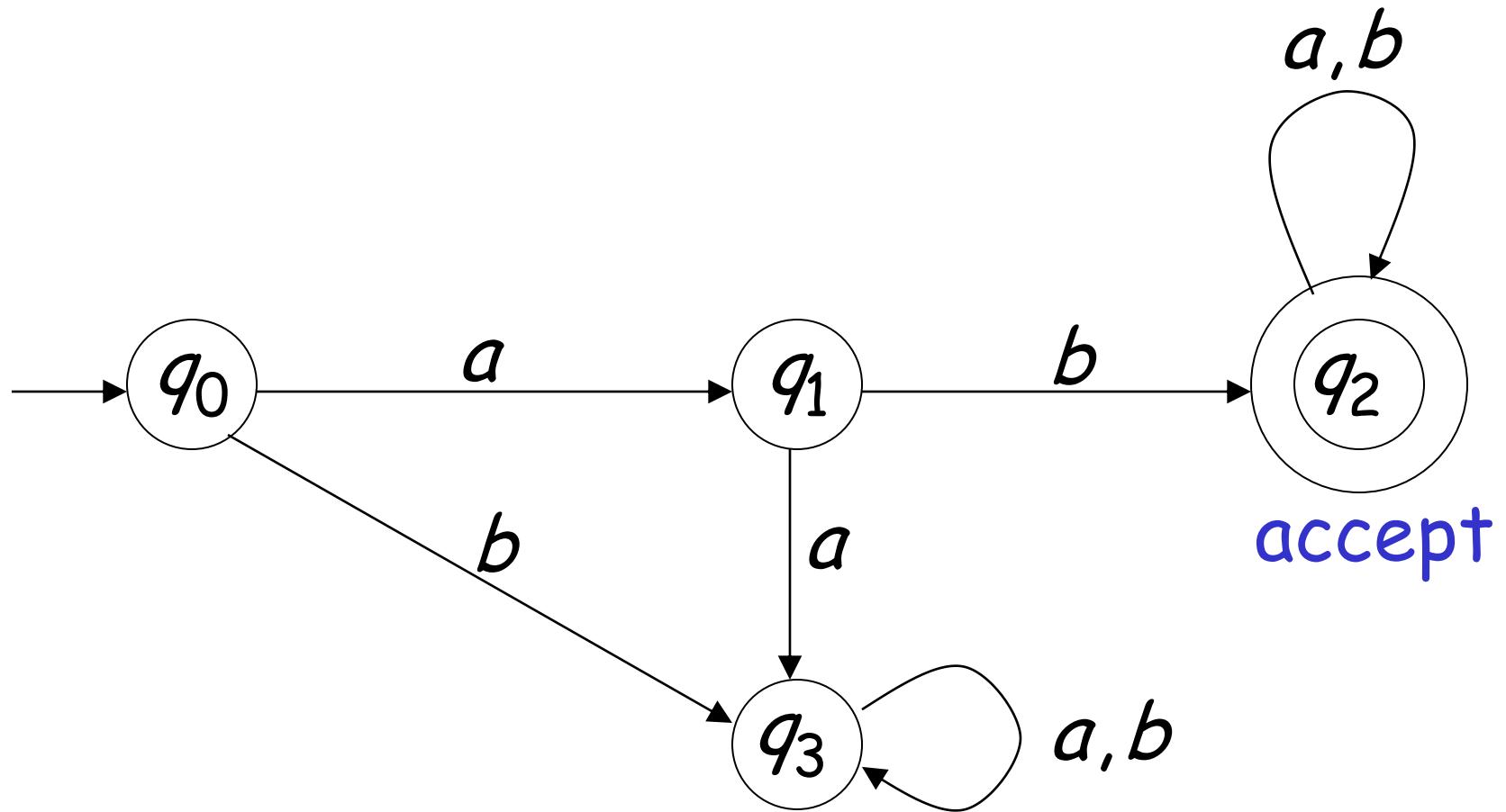
- Deterministic Finite State Machines
- $\{Q, \Sigma, q_0, A, \delta\}$
- Language = DFA (DFSM)
- $L = \text{Set of all strings started with symbol 'a' over the alphabet } \{a, b\}$
- $L = \{a, aa, ab, aab, aba, abb, \dots\}$
- String=aabab and baab
- Lexical Analyzer



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

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

$L = \{ ab, aba, abb, abaa, abba, abbb, abab, \dots \}$



# Language Accepted by DFA

Language of DFA  $M$  :

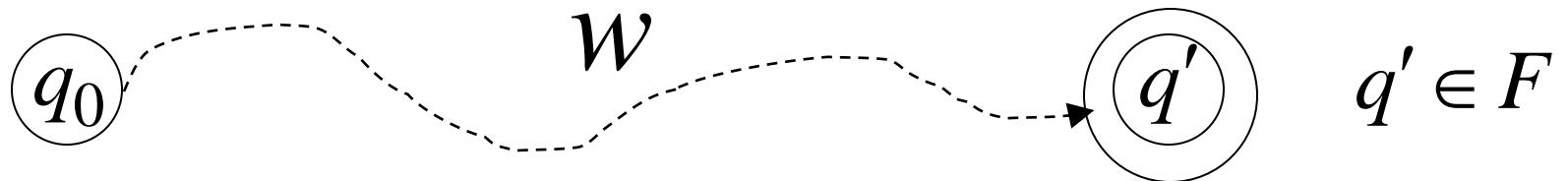
it is denoted as  $L(M)$  and contains all the strings accepted by  $M$

We say that a language  $L'$  is accepted (or recognized) by DFA  $M$  if  $L(M) = L'$

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

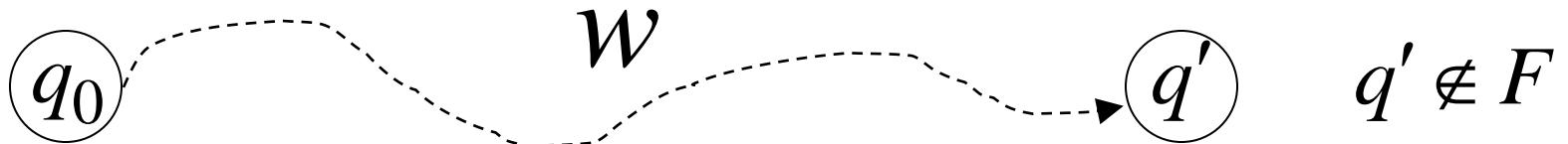
Language accepted by  $M$ :

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



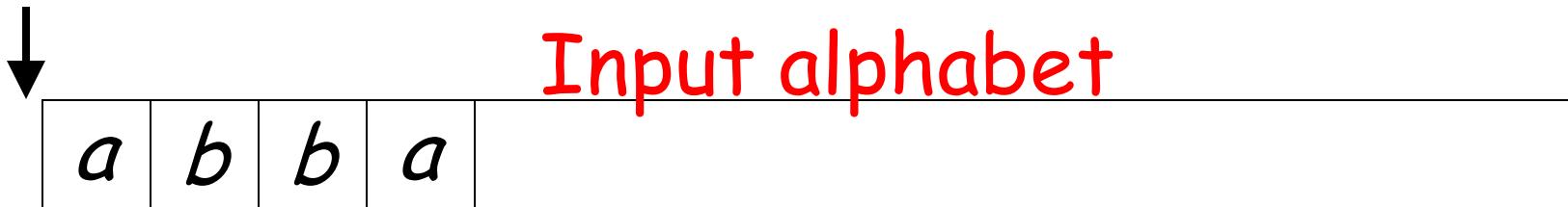
Language rejected by  $\mathcal{M}$ :

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

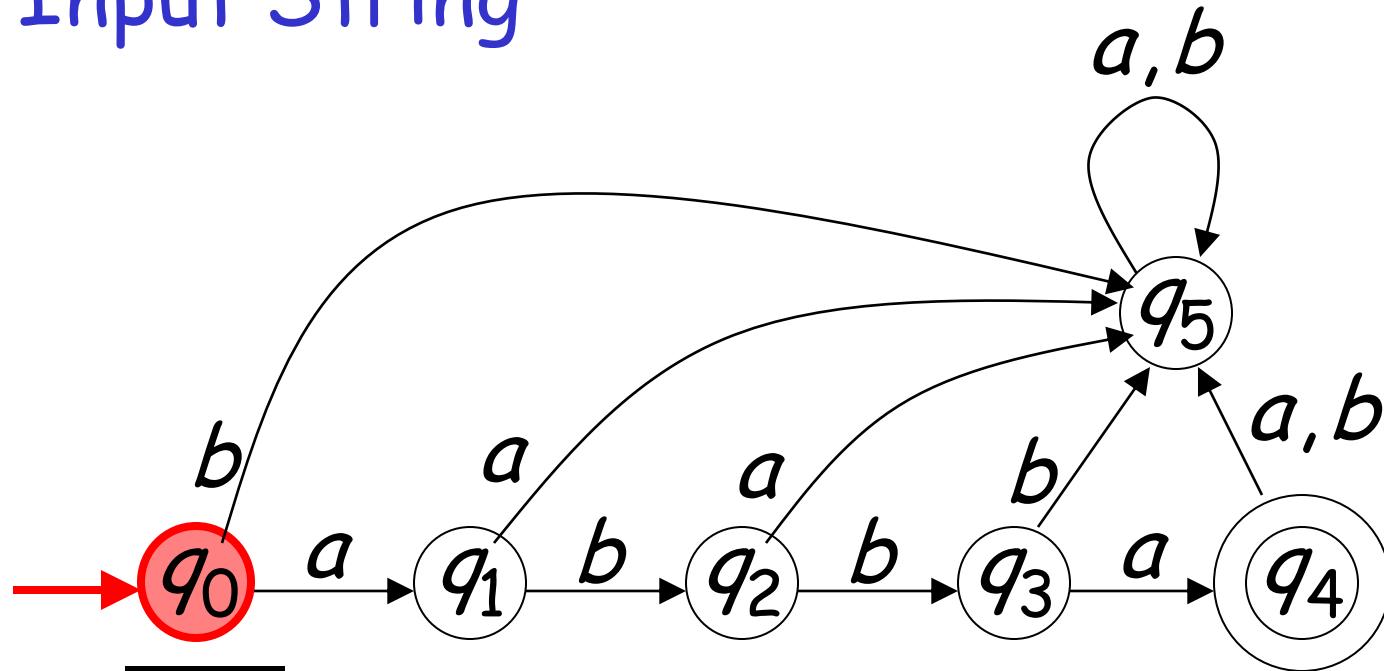


head

# Initial Configuration

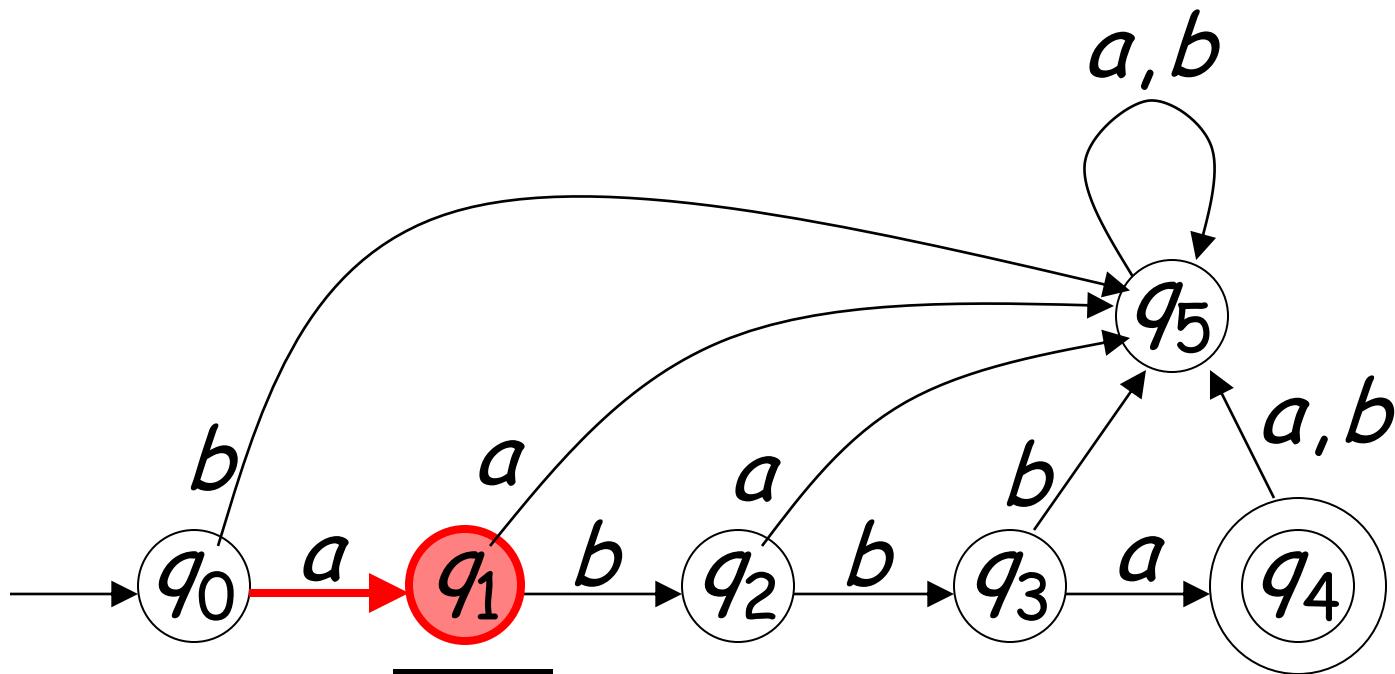
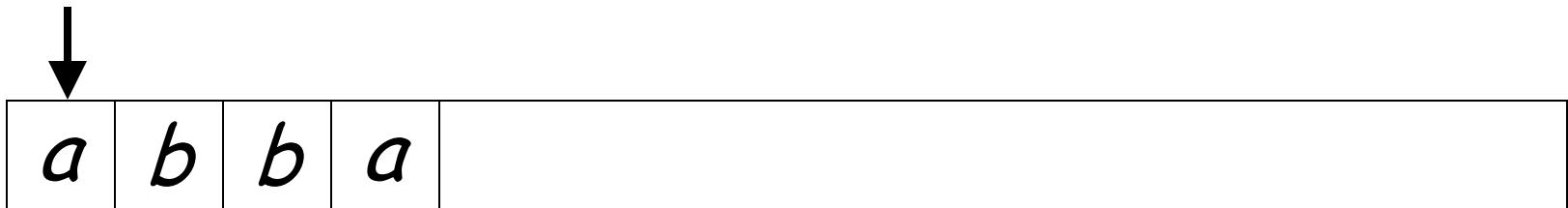


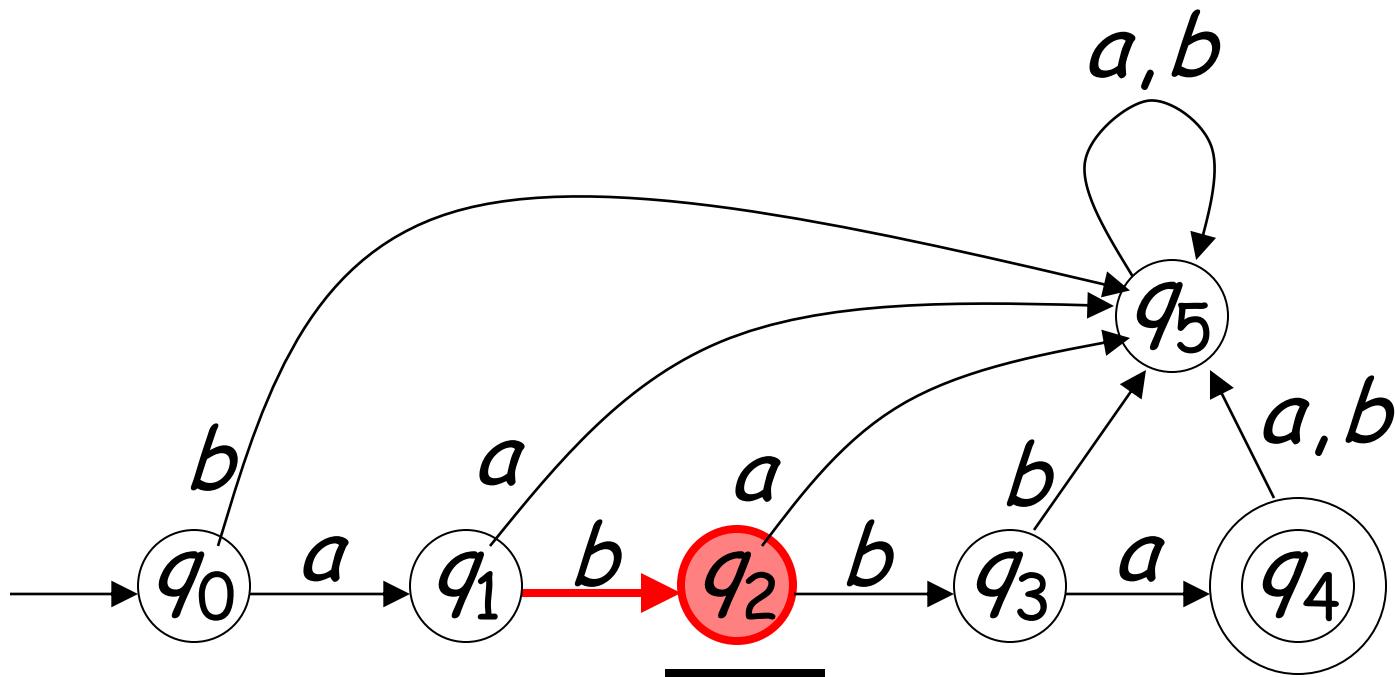
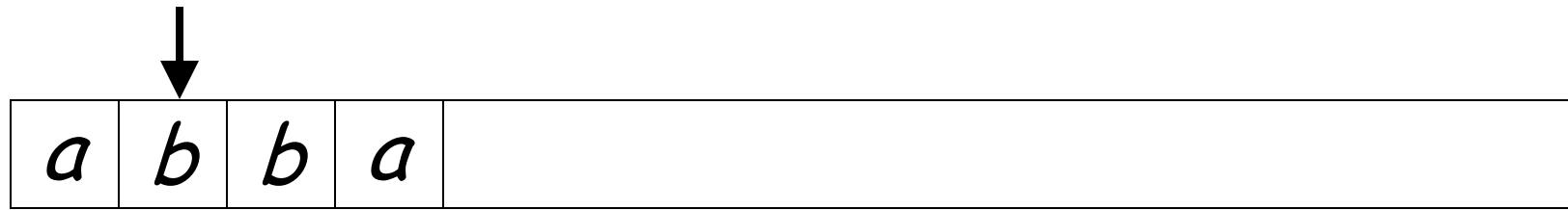
Input String

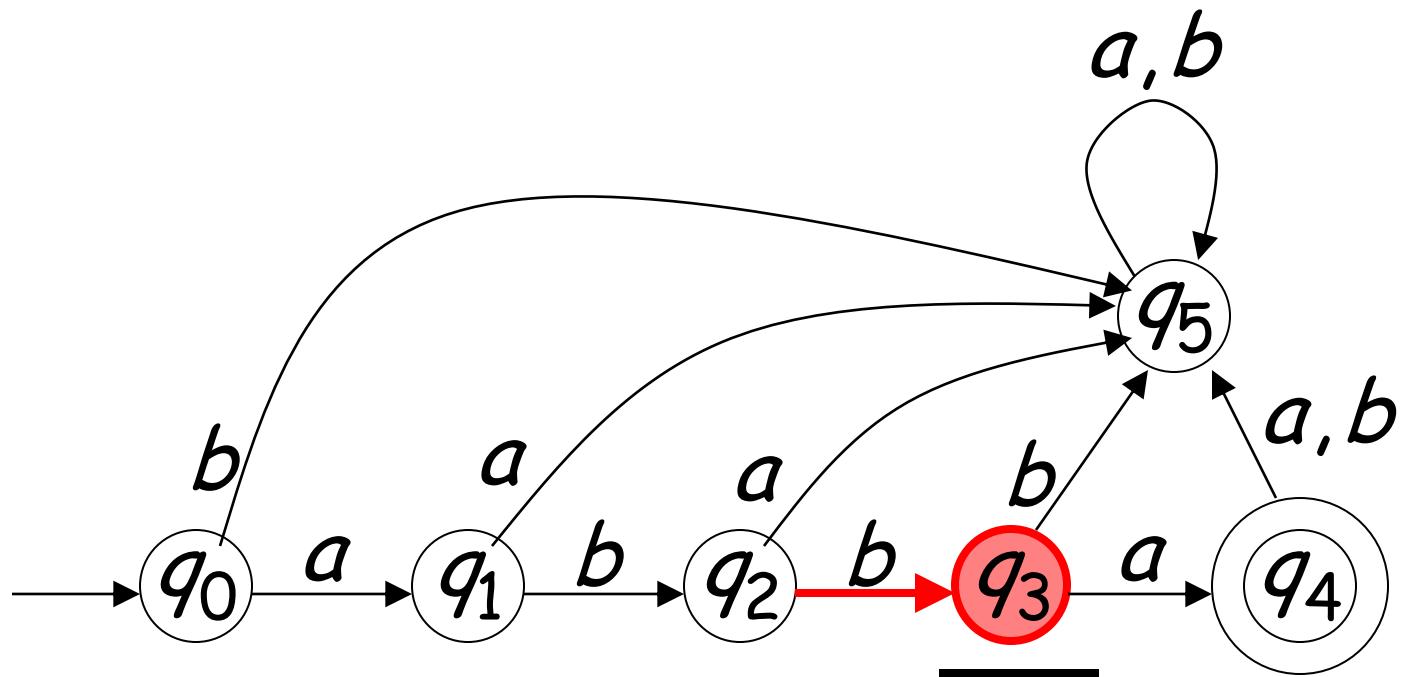
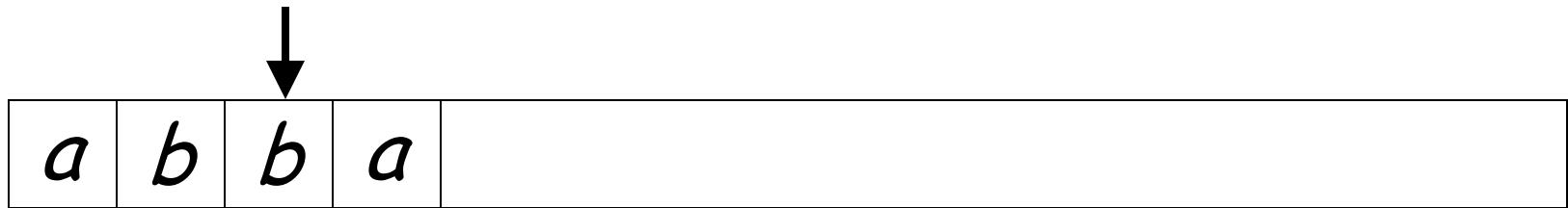


Initial state

# Scanning the Input



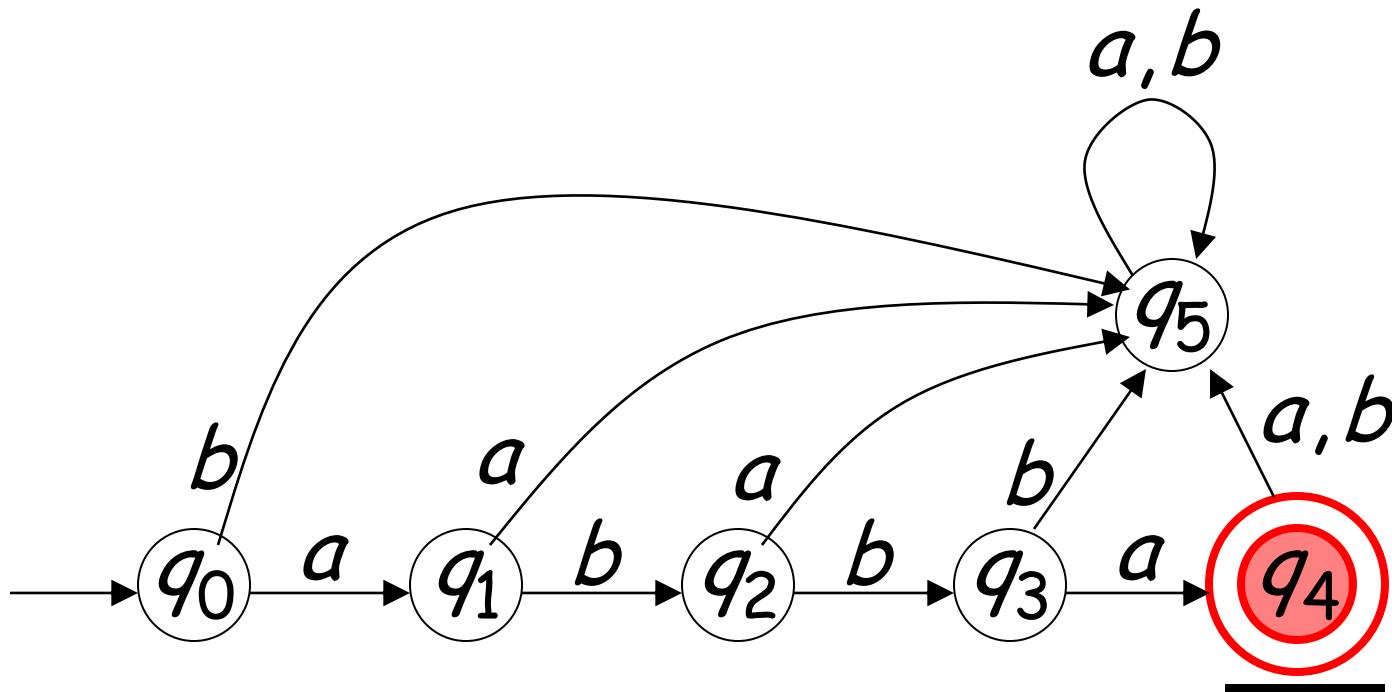




Input finished



a	b	b	a	
---	---	---	---	--



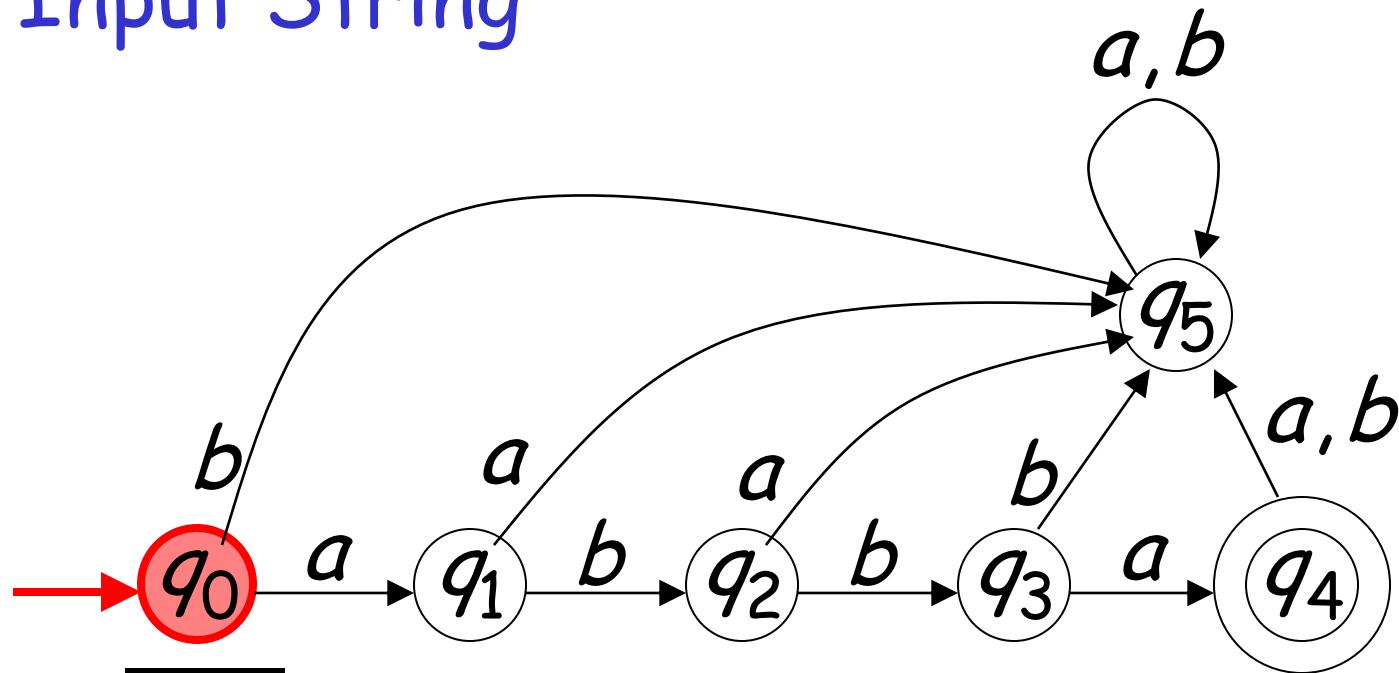
accept

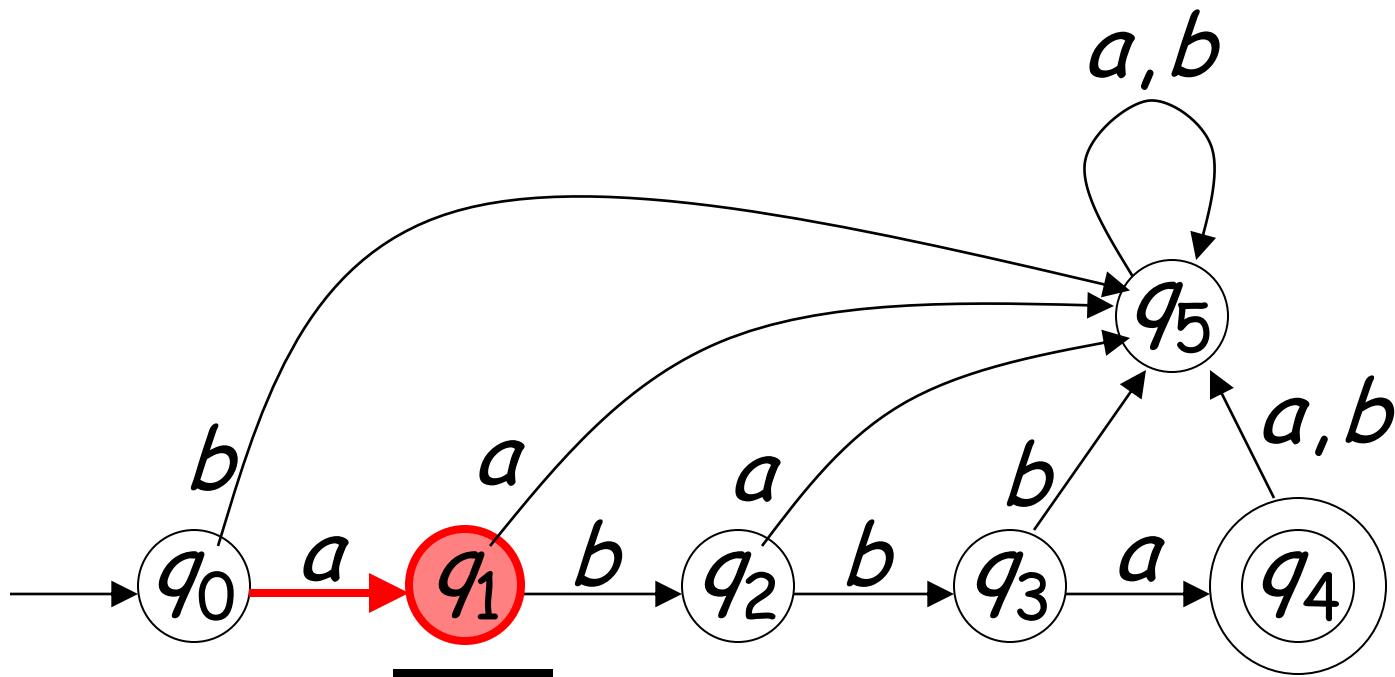
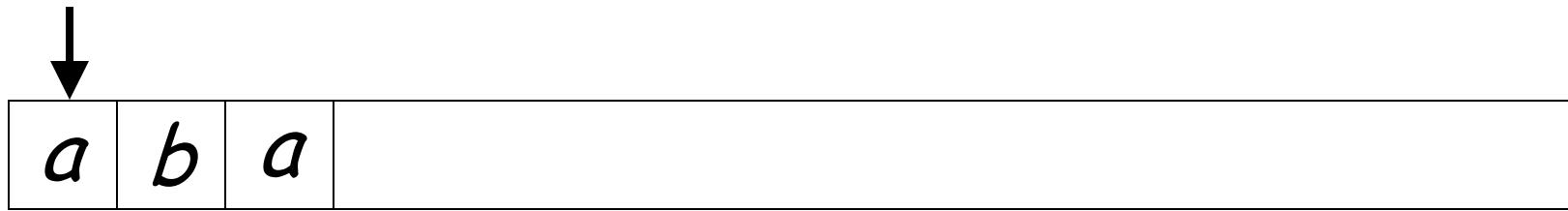
# A Rejection Case



a	b	a	
---	---	---	--

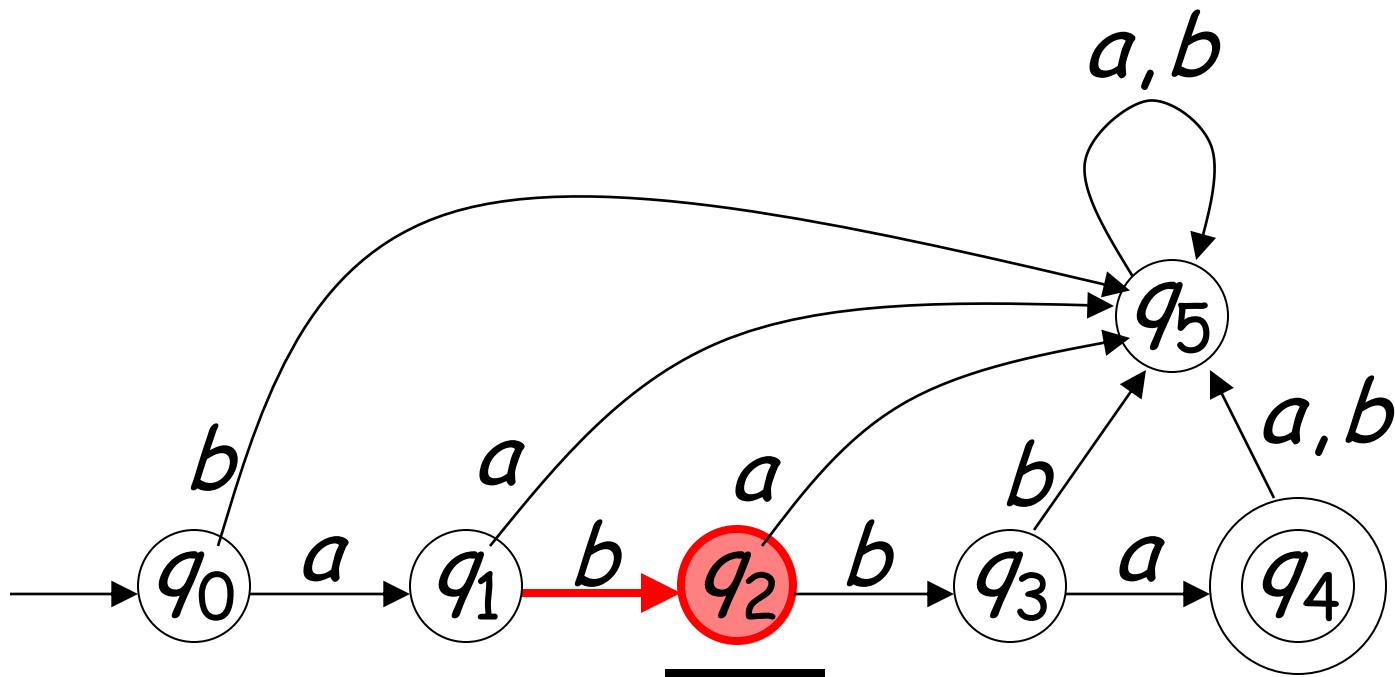
Input String





↓

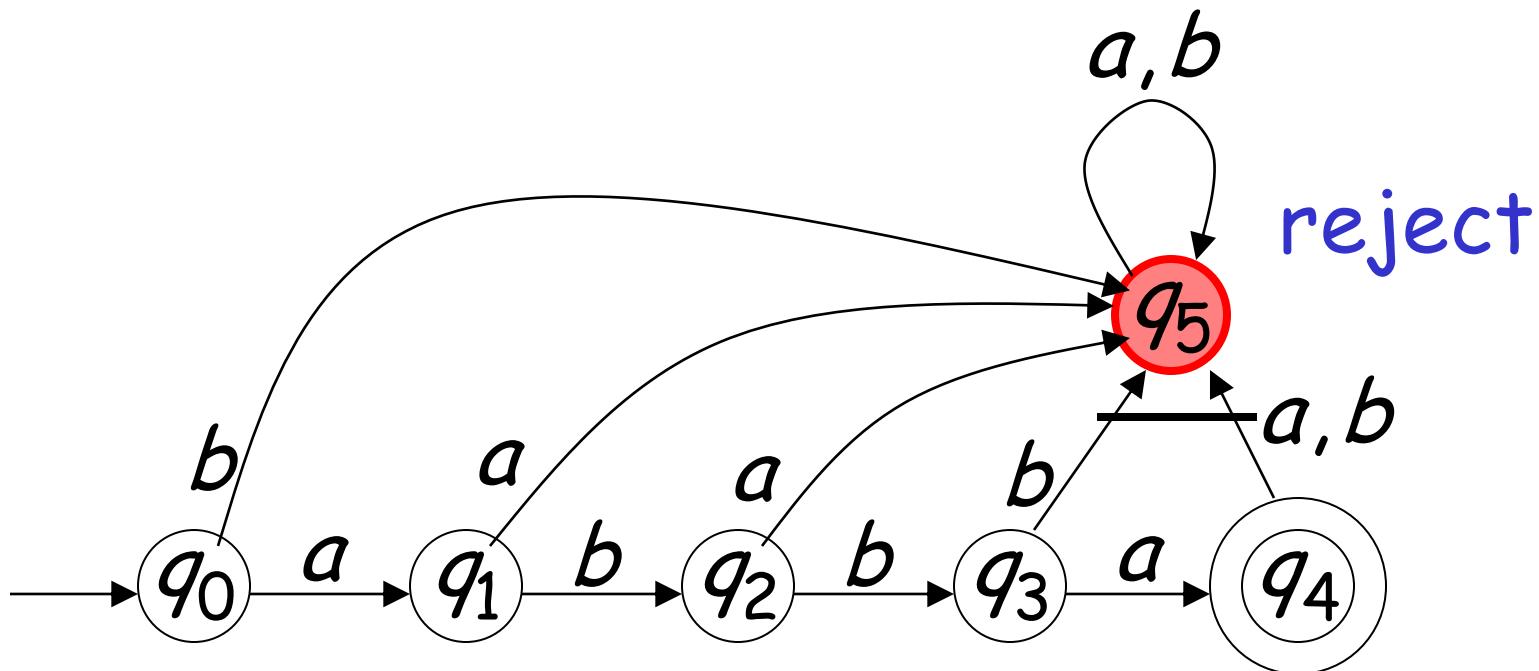
$a$	$b$	$a$	
-----	-----	-----	--



Input finished



a	b	a	
---	---	---	--

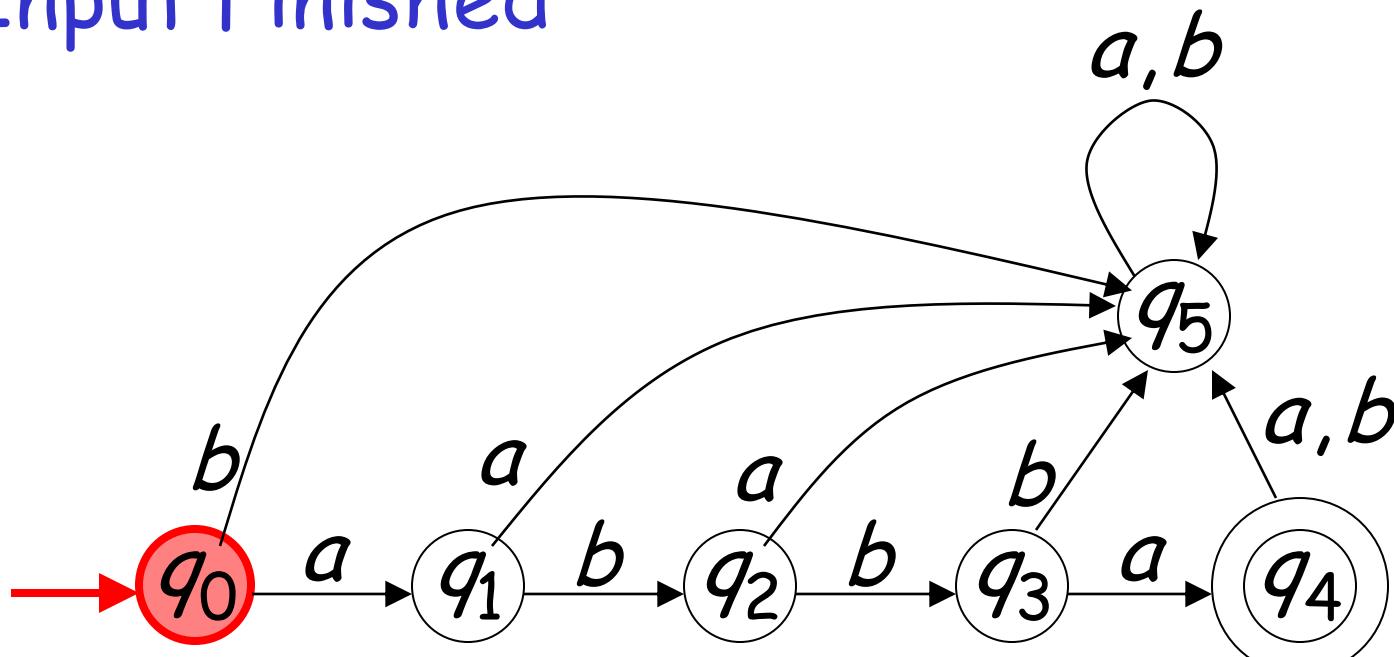


# Another Rejection Case

Input String is empty

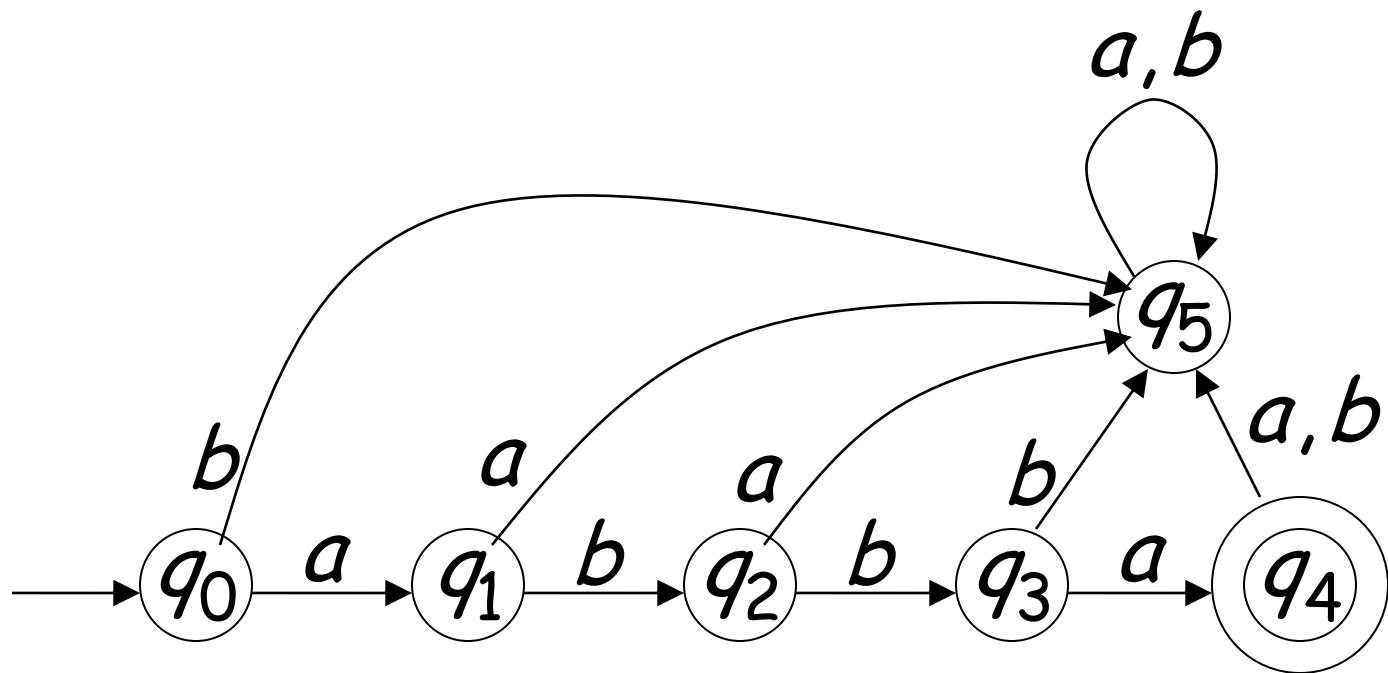
$(\lambda)$

Input Finished



reject

Language Accepted:  $L = \{abba\}$



To accept a string:

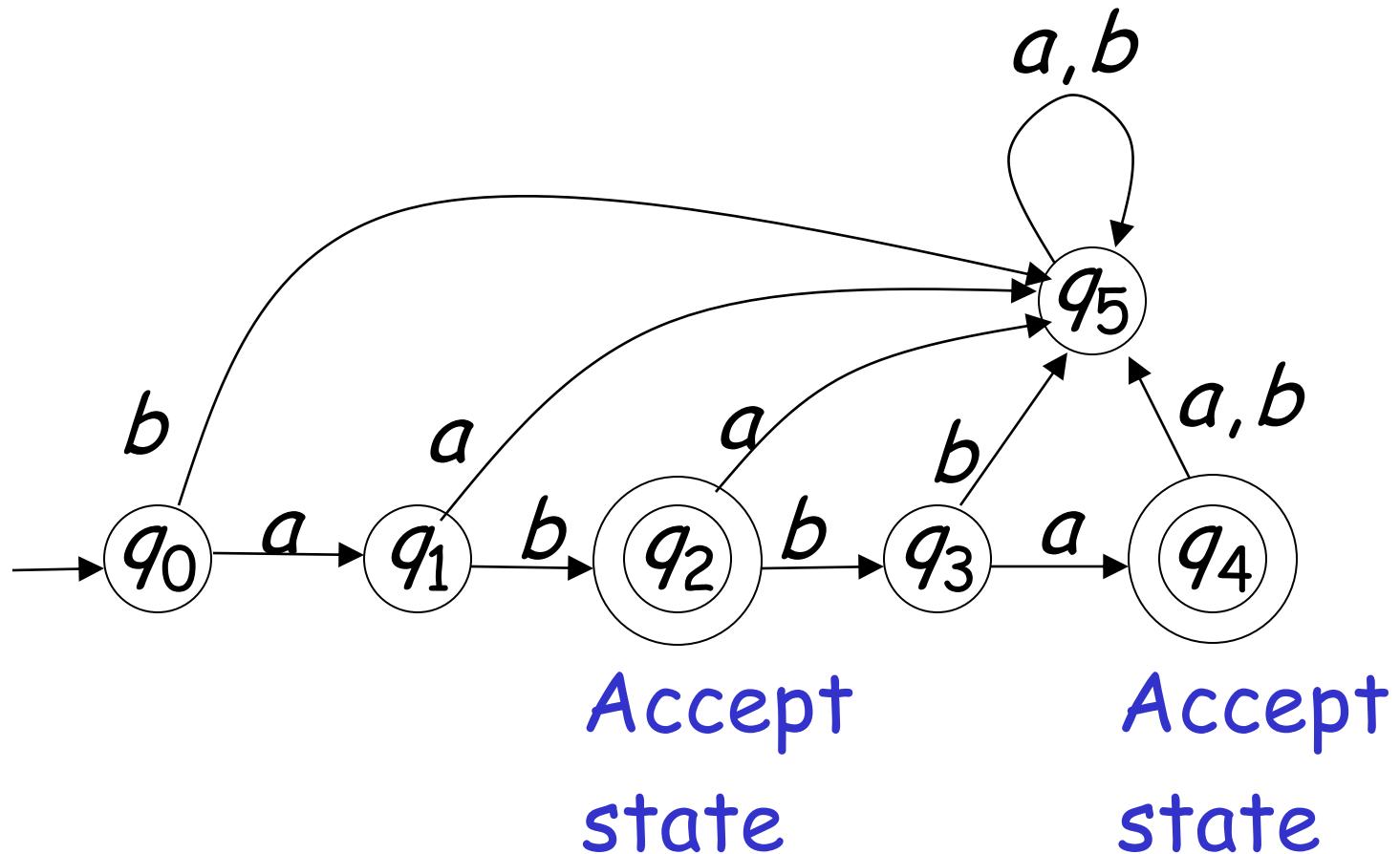
all the input string is scanned  
and the last state is accepting

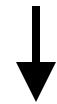
To reject a string:

all the input string is scanned  
and the last state is non-accepting

# Another Example

$$L = \{ab, abba\}$$

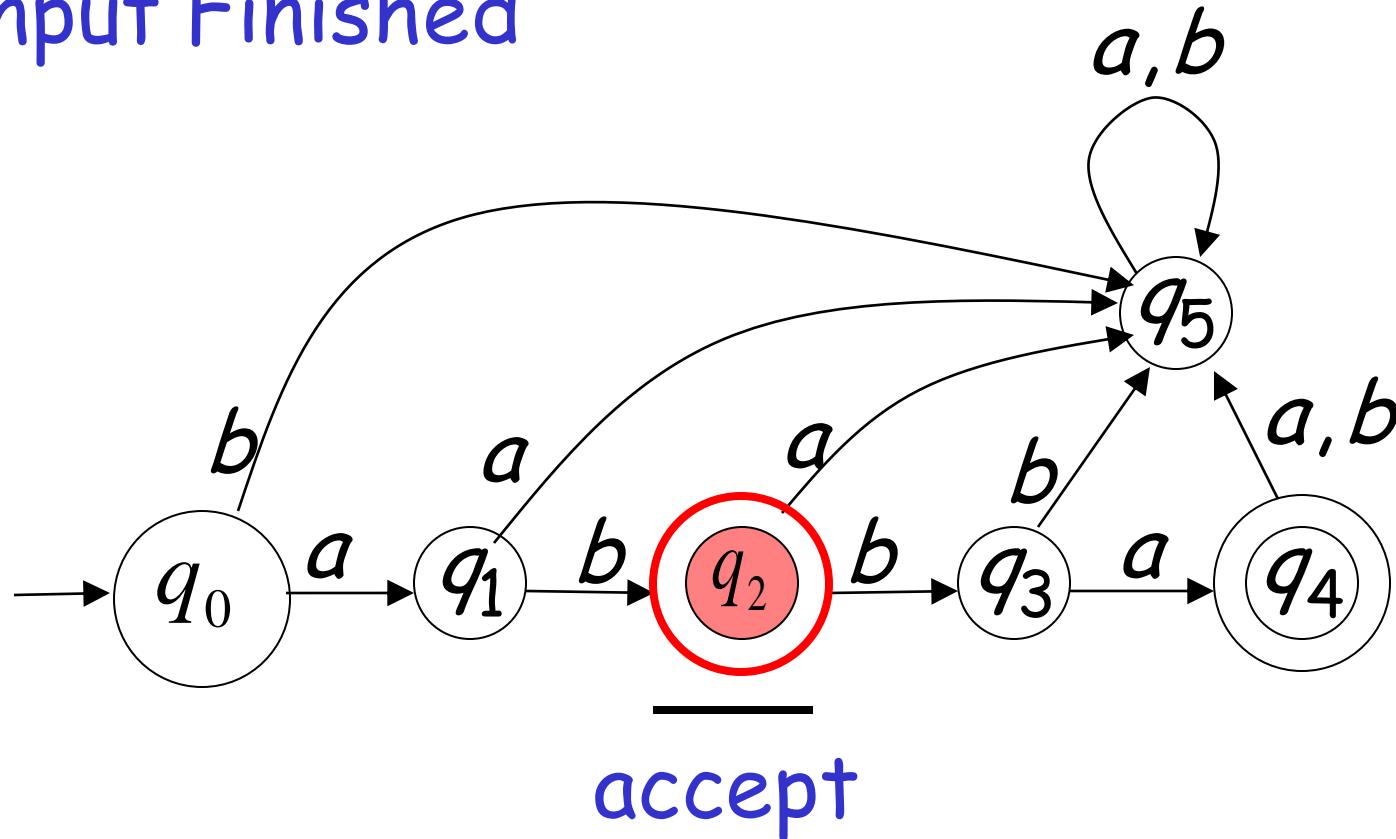




Input Tape

$ab$

Input Finished



# Language Accepted by DFA

Language of DFA  $M$ :

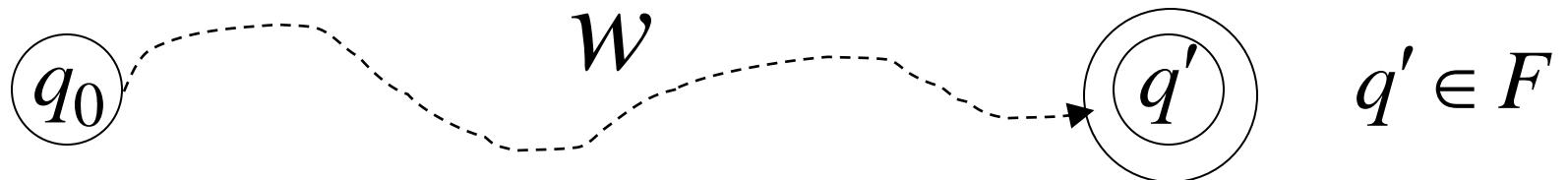
it is denoted as  $L(M)$  and contains all the strings accepted by  $M$

We say that a language  $L'$  is accepted (or recognized) by DFA  $M$  if  $L(M) = L'$

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

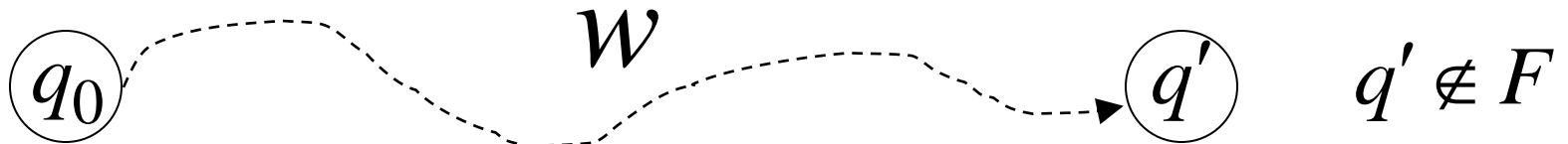
Language accepted by  $M$ :

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

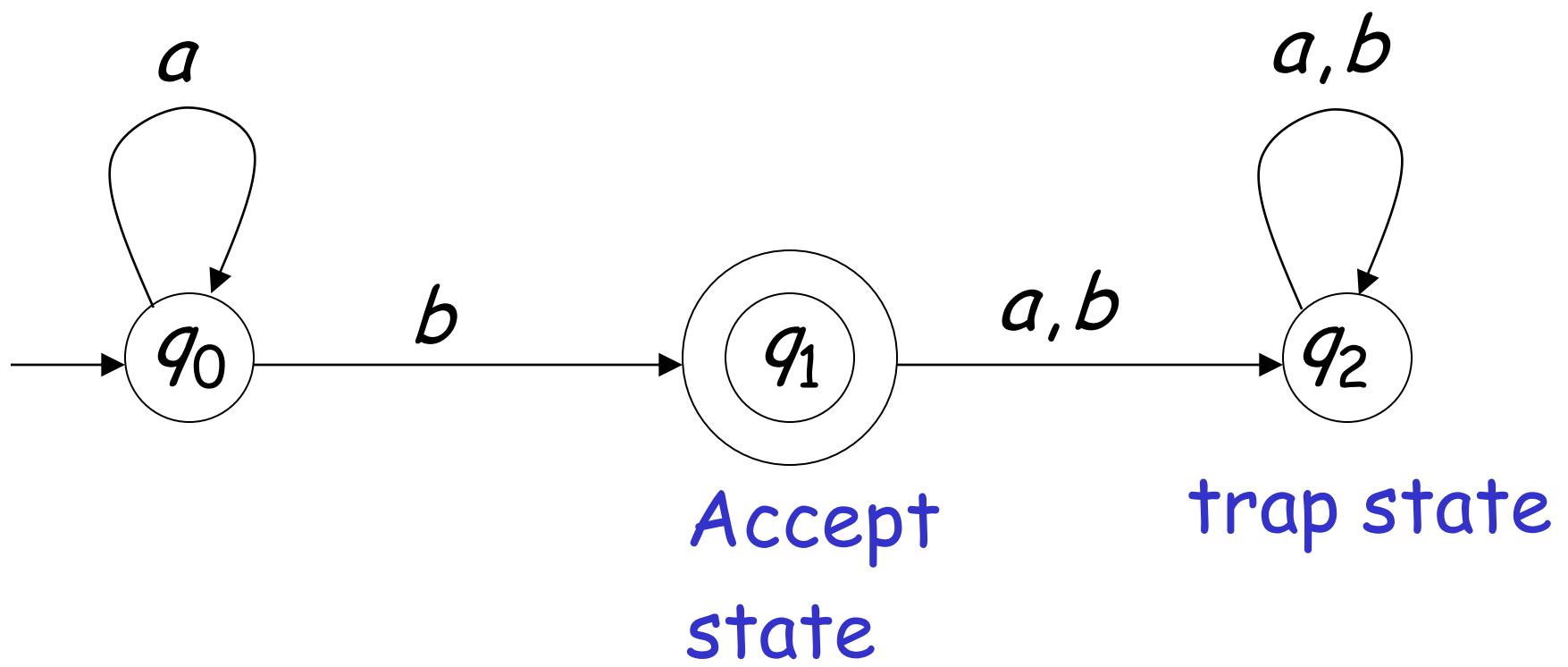


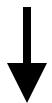
Language rejected by  $\mathcal{M}$ :

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



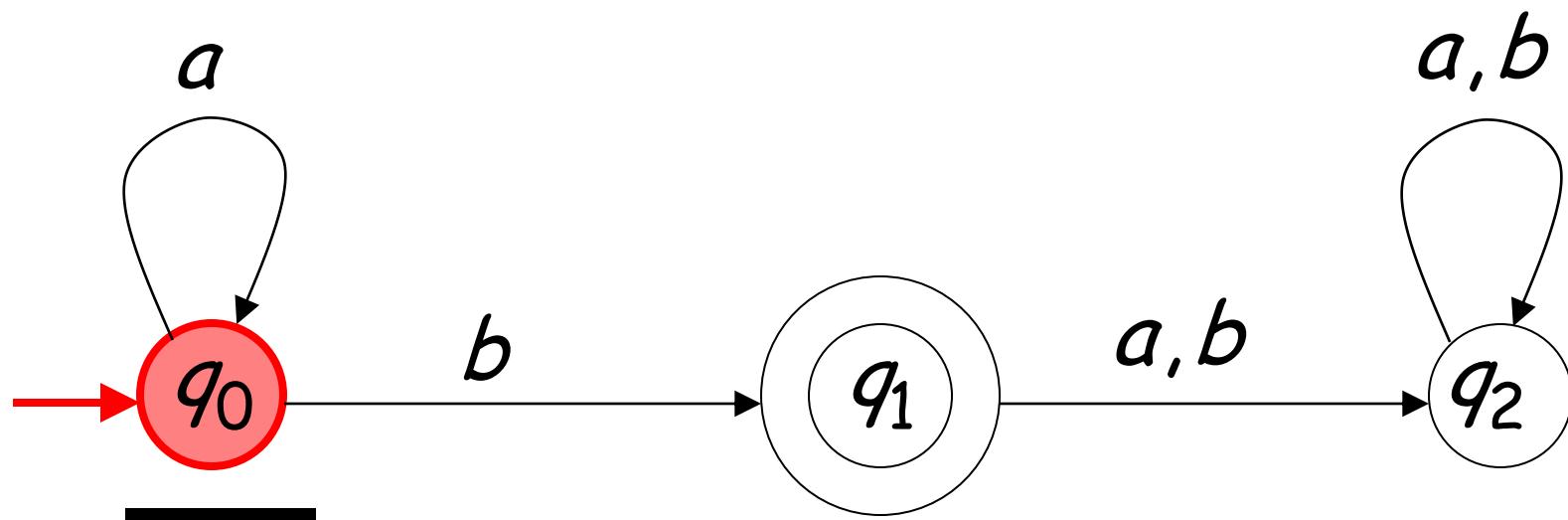
# Another Example





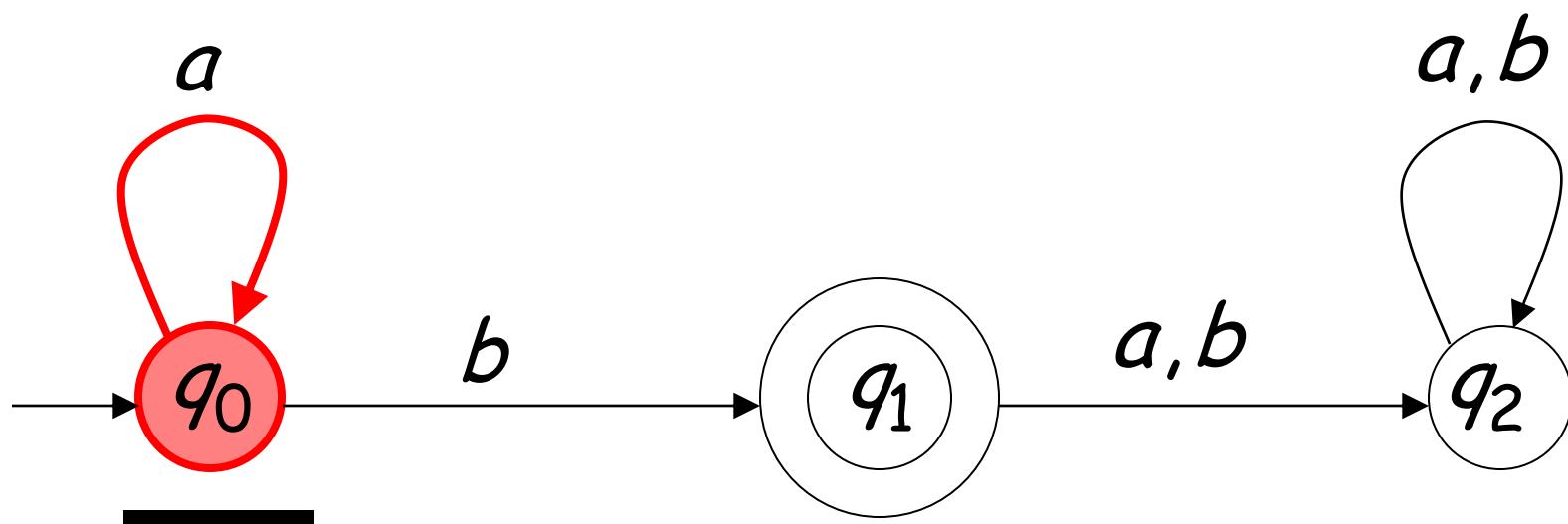
$a$	$a$	$b$	
-----	-----	-----	--

Input String



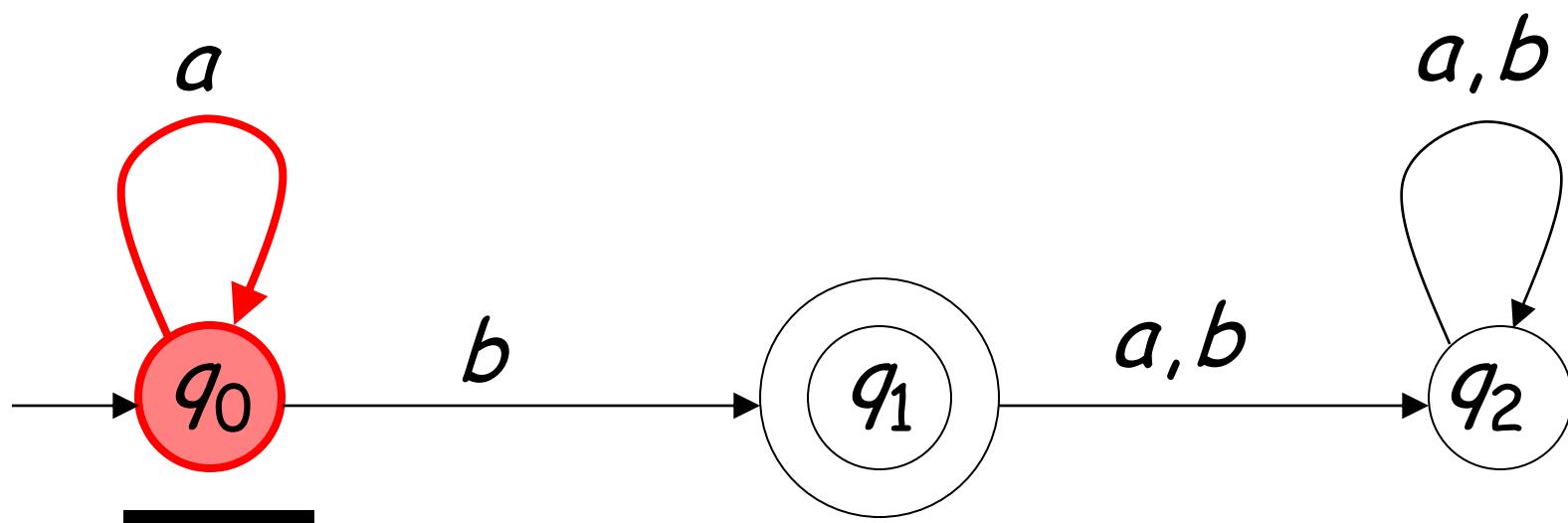
↓

$a$	$a$	$b$	
-----	-----	-----	--

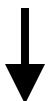


↓

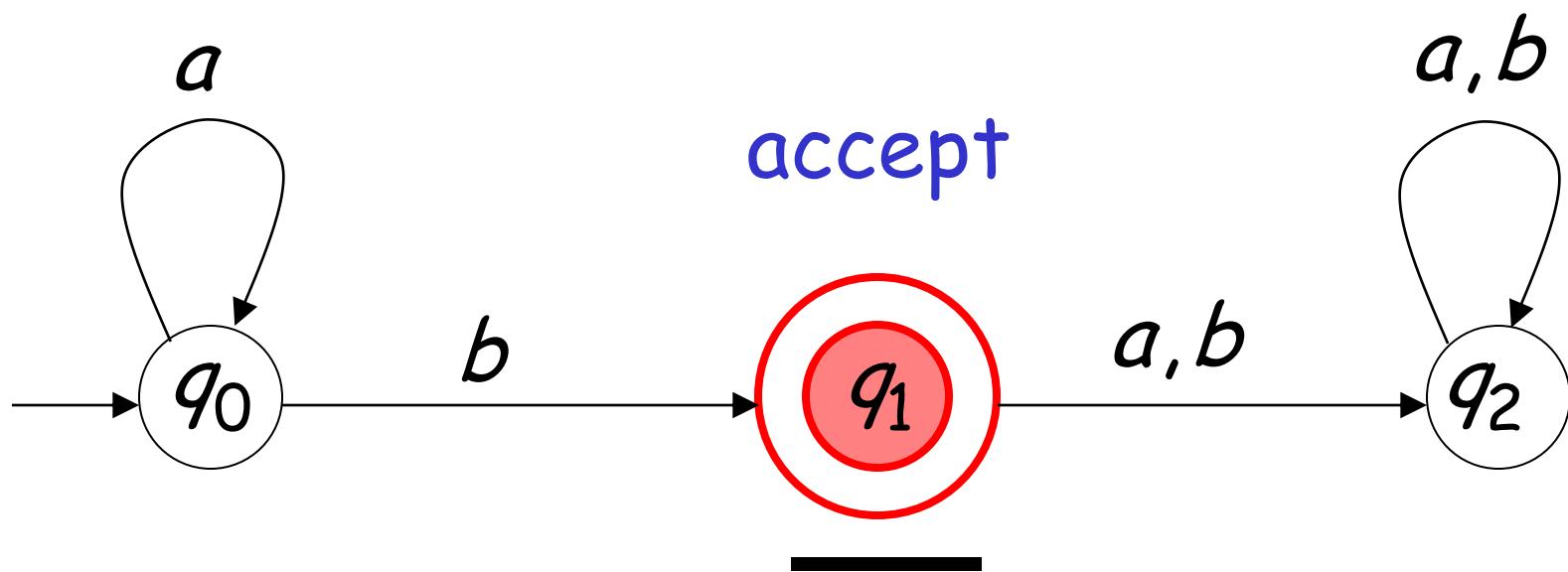
$a$	$a$	$b$	
-----	-----	-----	--



# Input finished



a	a	b	
---	---	---	--

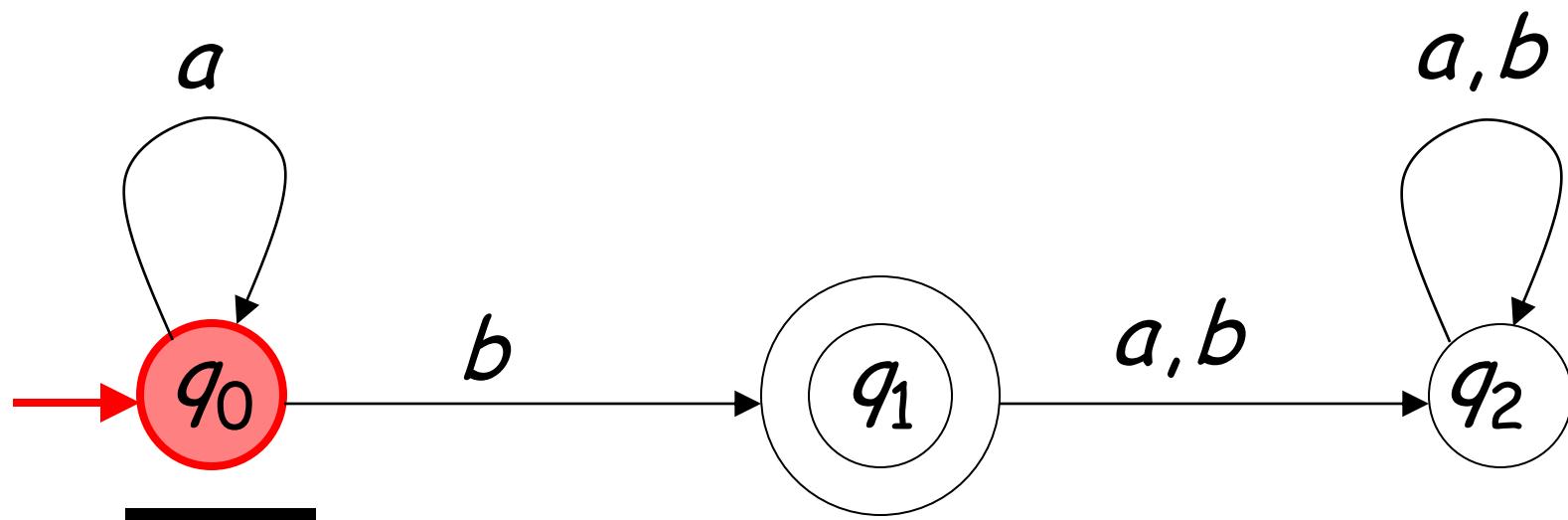


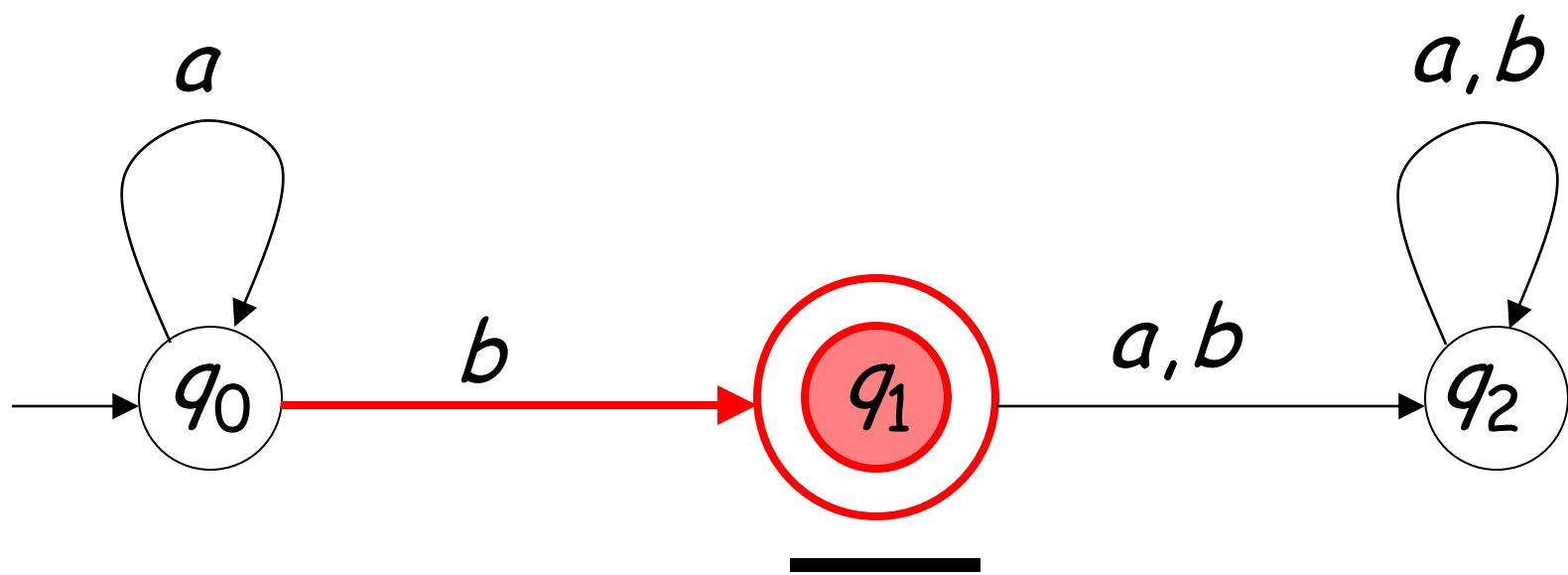
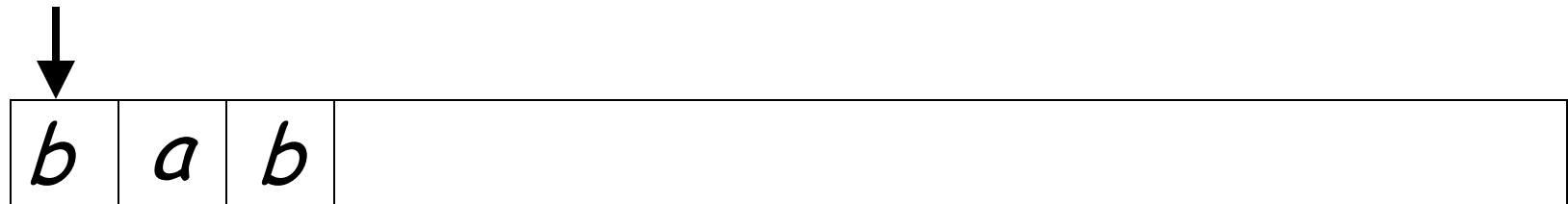
# A rejection case

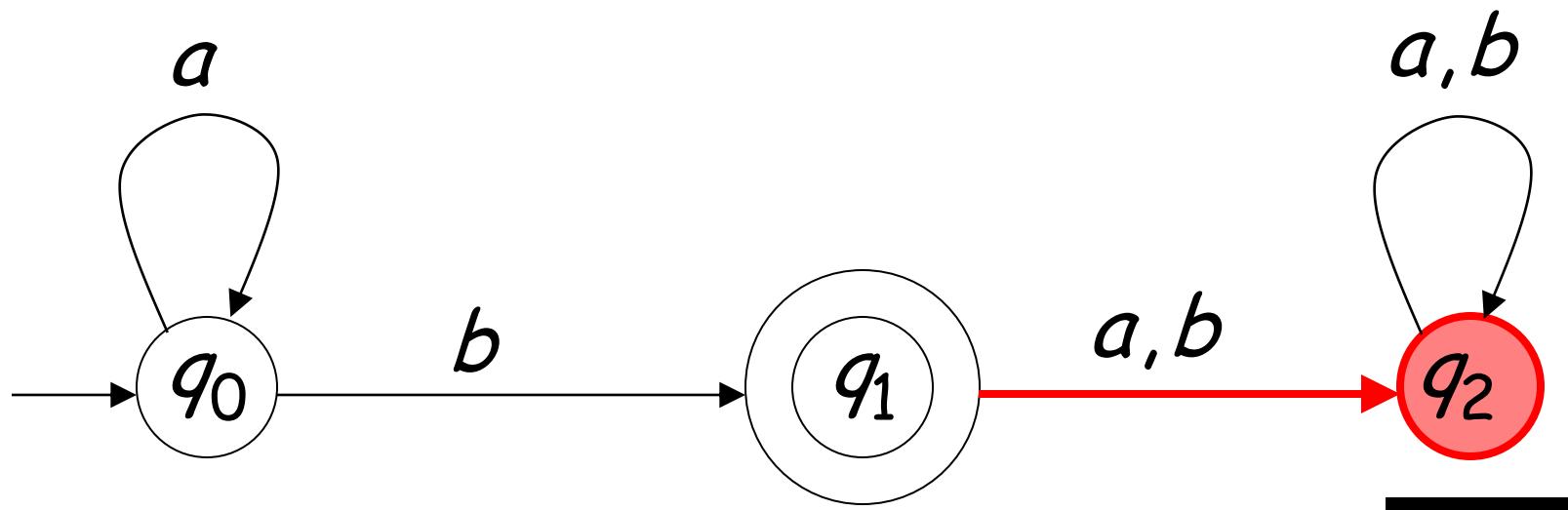
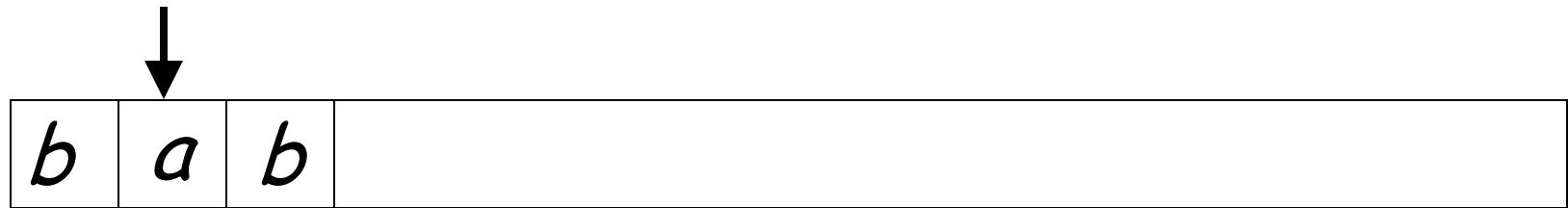


b	a	b	
---	---	---	--

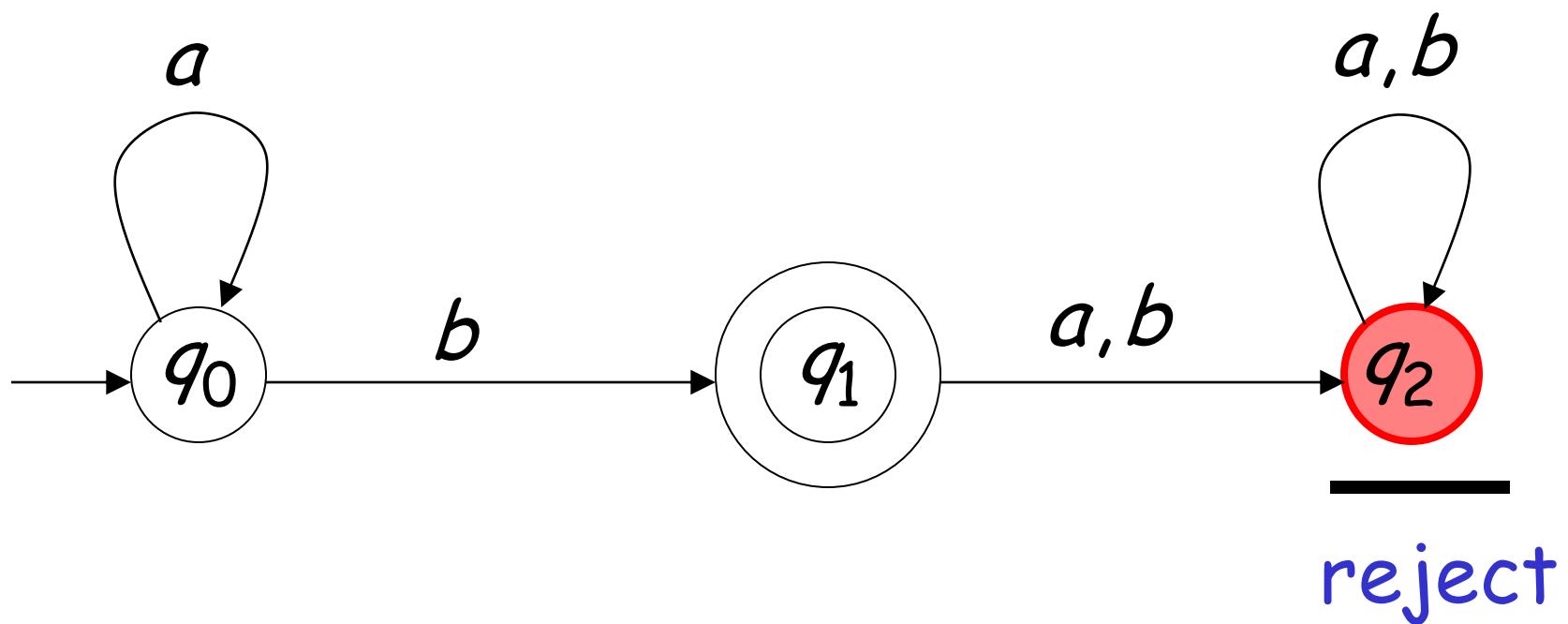
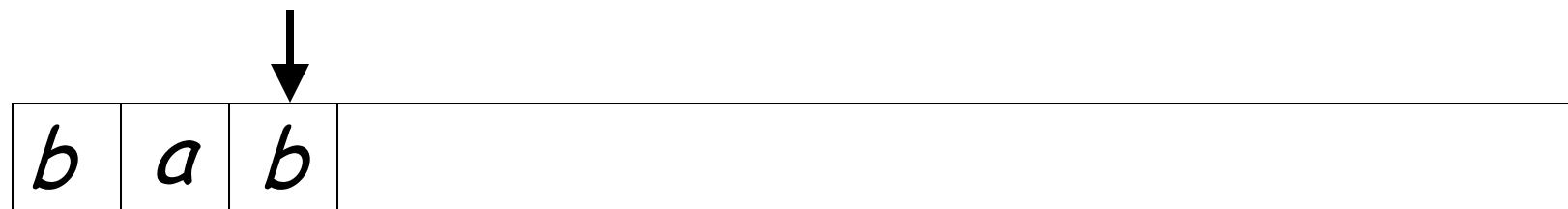
Input String



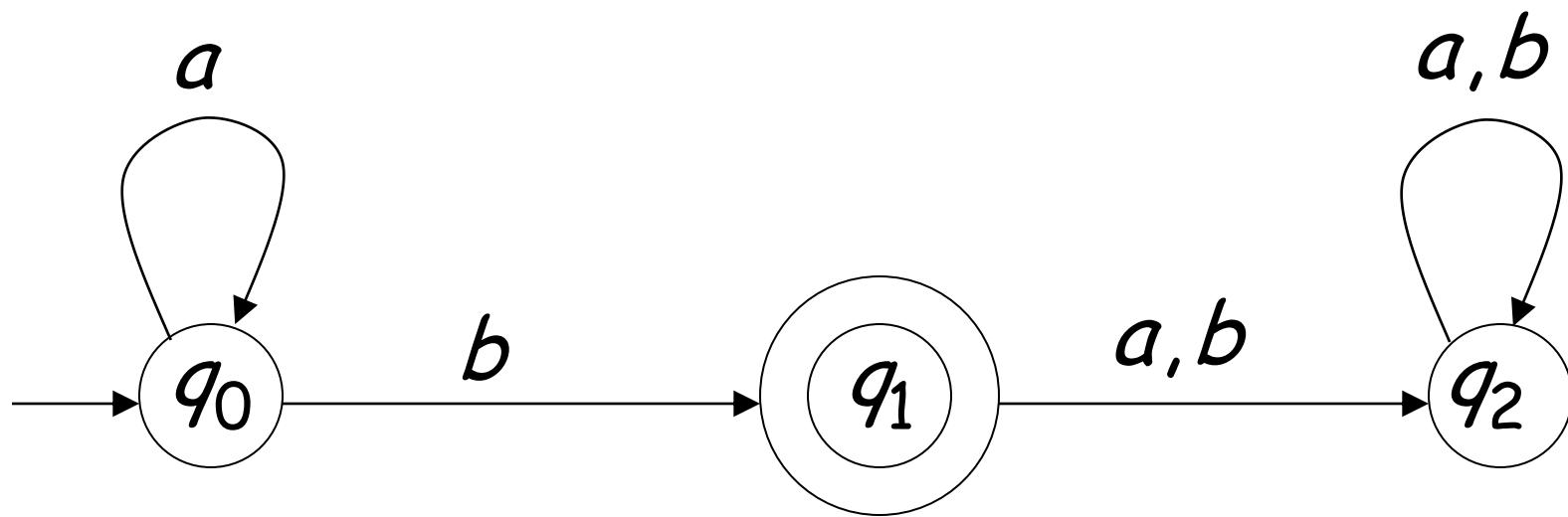




Input finished

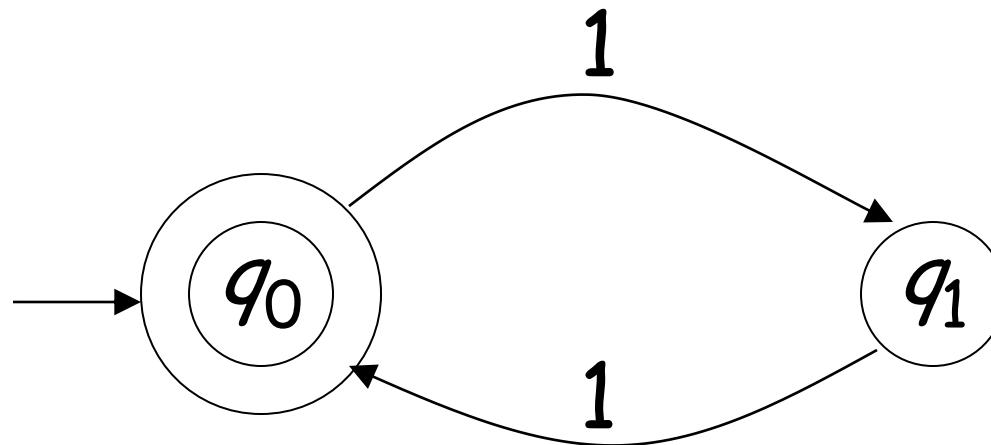


Language Accepted:  $L = \{a^n b : n \geq 0\}$



# Another Example

Alphabet:  $\Sigma = \{1\}$



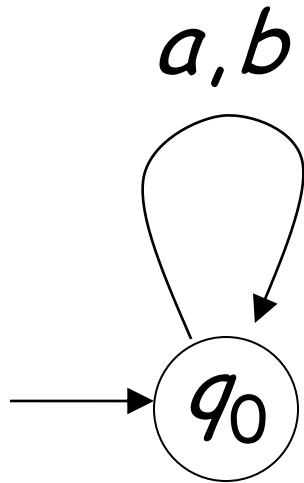
Language Accepted:

$$EVEN = \{x : x \in \Sigma^* \text{ and } x \text{ is even}\}$$

$$= \{\lambda, 11, 1111, 111111, \dots\}$$

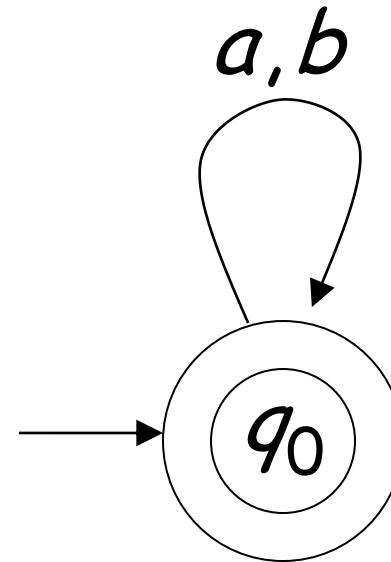
# More DFA Examples

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



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

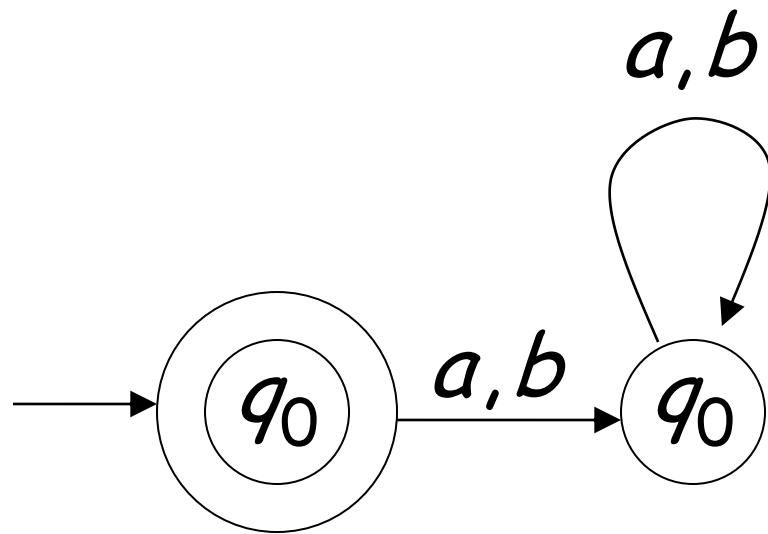
Empty language



$$L(M) = \Sigma^*$$

All strings

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

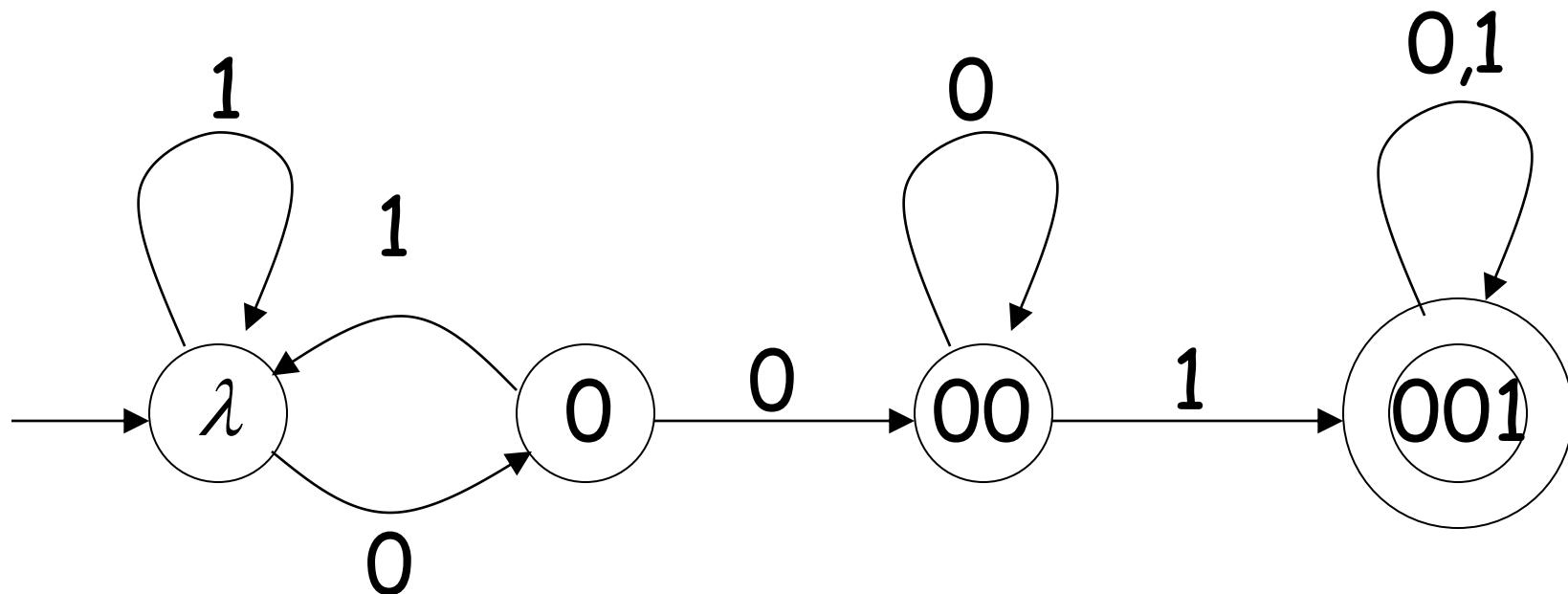


$$L(M) = \{\lambda\}$$

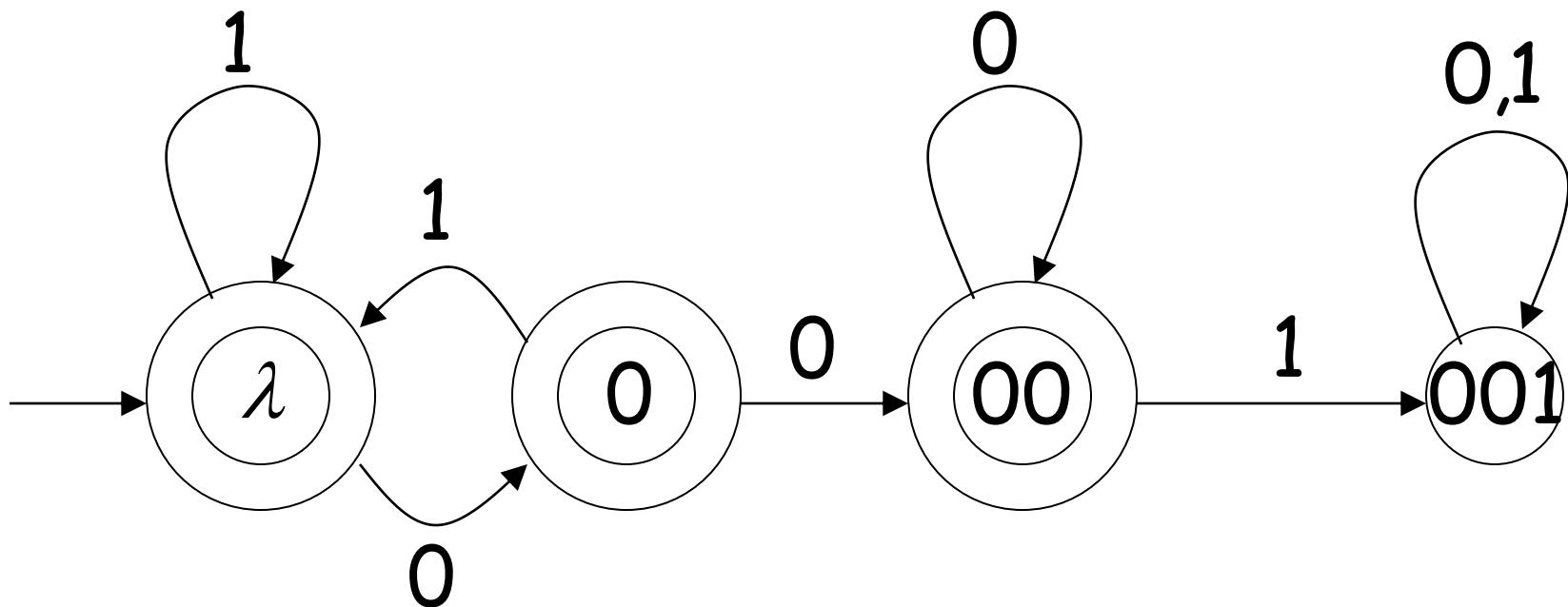
Language of the empty string

$L(M) = \{ \text{ all binary strings containing substring } 001 \ }$

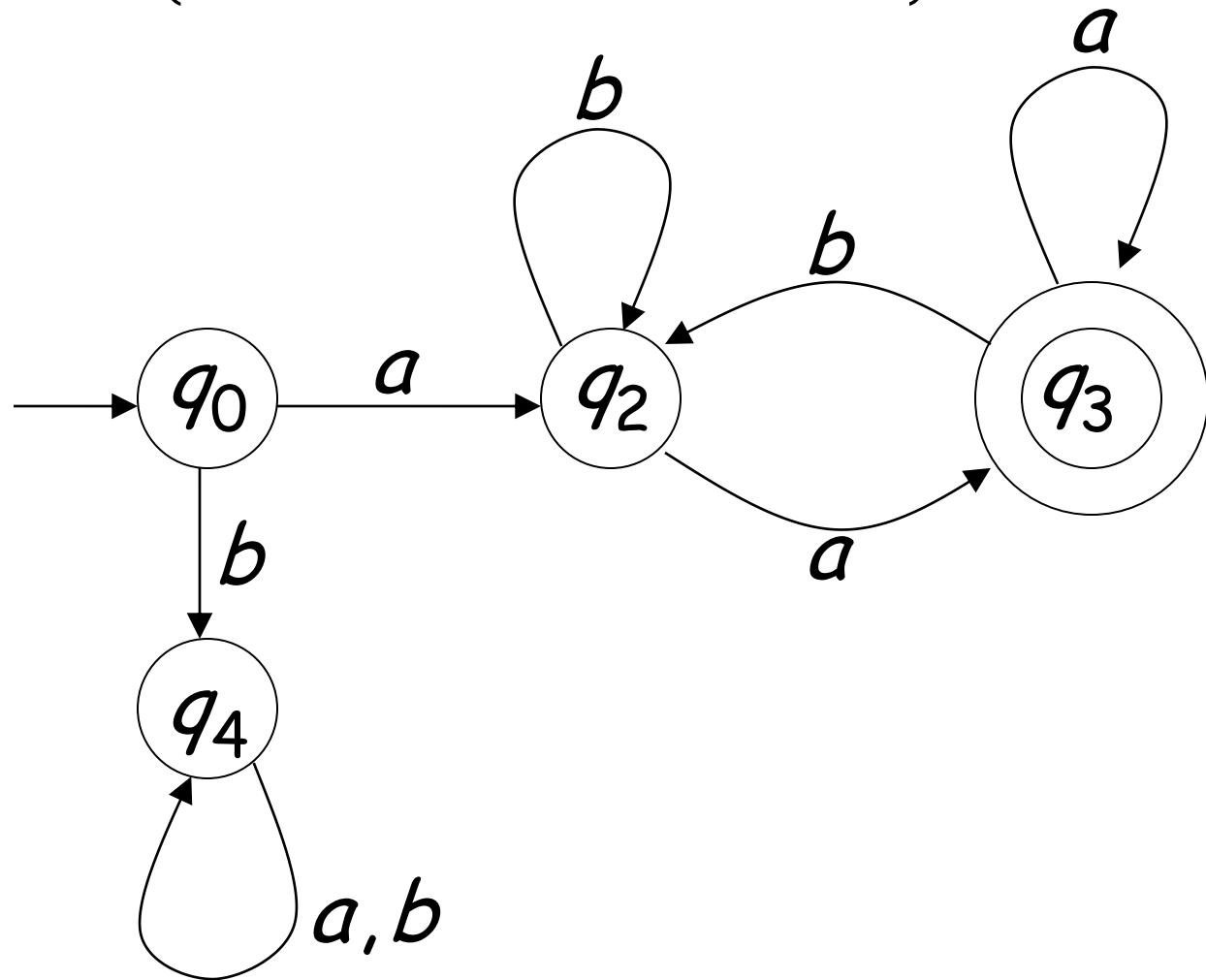
$L = \{ 001, 1001, 0010, 0011, 00001, \dots \}$



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



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



# Exercise

- A Finite Automaton Accepting the Language of Strings Ending in aa over the alphabet {a,b}
- An FA accepting the strings ending with *b* and *not containing aa*.
- An FA Accepting Binary Representations of Integers Divisible by 3
- An FA Accepting Strings That Contain Either ab or bba.

- FA which read string made of {0,1} and accepts those string which end with 00 or 11.(Insem-2017, 4 Marks)
- Design FA Accepting Language of Strings Ending in b and not containing the substring aa.
- An FA accepting Binary Representations of Integers Divisible by 3.

# Regular Languages

Definition:

A language  $L$  is regular if there is a DFA  $M$  that accepts it ( $L(M) = L$ )

The languages accepted by all DFAs form the family of regular languages

**A language  $L$  is **regular** if it is  
**recognized** by a deterministic  
finite automaton (DFA),  
i.e. if there is a DFA  $M$  such  
that  $L = L(M)$ .**

$L = \{ w \mid w \text{ contains } 001\}$  is regular

$L = \{ w \mid w \text{ has an even number of } 1's\}$  is regular

## Example regular languages:

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

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

{ all strings in  $\{a,b\}^*$  with prefix  $ab$  }

{ all binary strings without substring 001}

{ $x : x \in \{1\}^*$  and  $x$  is even}

{ }     $\{\lambda\}$      $\{a,b\}^*$

There exist automata that accept these languages (see previous slides).

There exist languages which are not Regular:

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

$$\text{ADDITION} = \{x + y = z : x = 1^n, y = 1^m, z = 1^k, n + m = k\}$$

There is no DFA that accepts these languages  
(we will prove this in a later class)

# Recall

- Examples of DFA
- NFA

# THEORY OF COMPUTATION

---

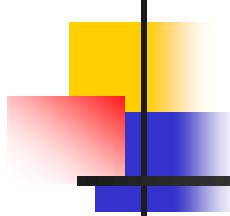
Unit I

## FINITE STATE MACHINES

---

**Jagadish Kashinath Kamble** (M.Tech, IIT Kharagpur)

Dept. of Information Technology,  
Pune Institute of Computer Technology, Pune



# Recall

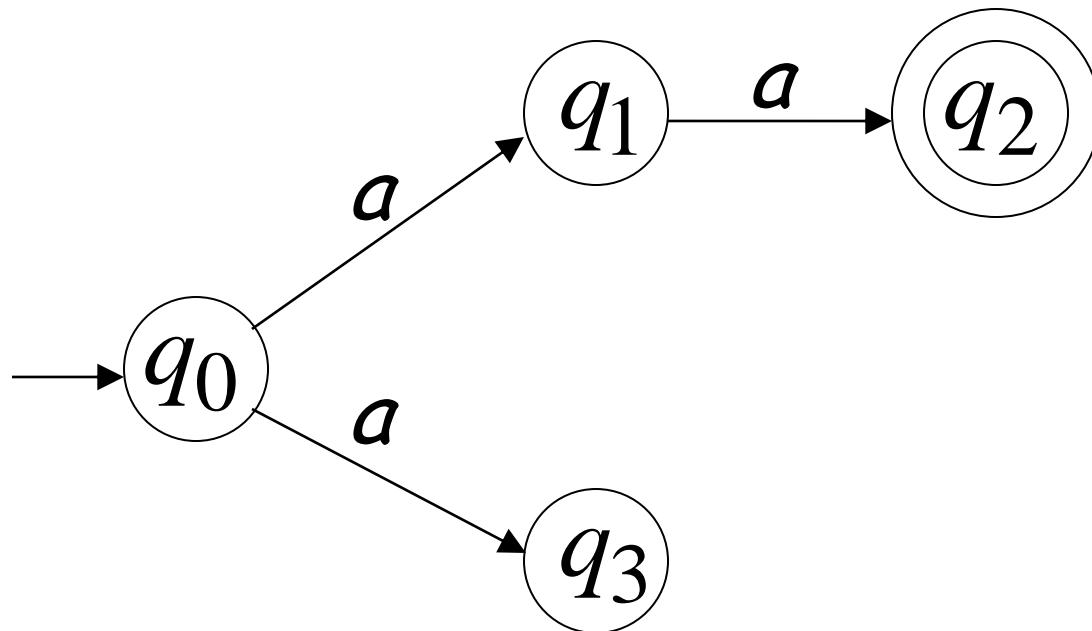
---

- Examples of DFA
- Regular Language

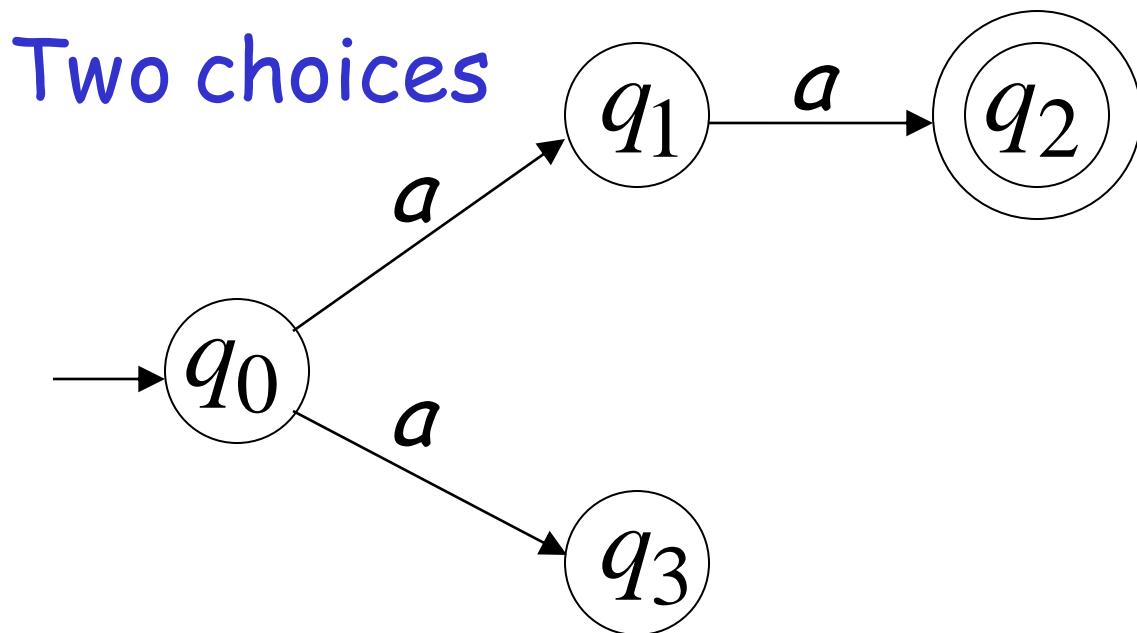
# Non-Deterministic Finite Automata

# Nondeterministic Finite Automaton (NFA)

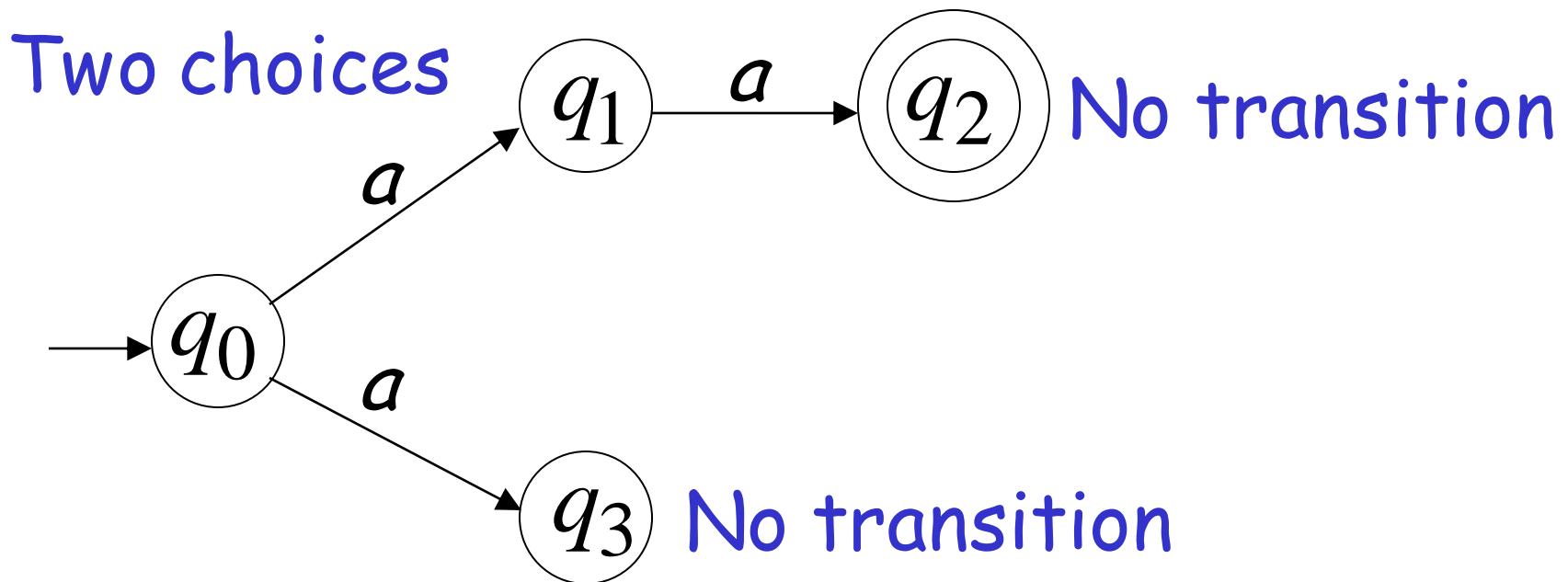
Alphabet =  $\{a\}$



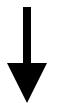
Alphabet =  $\{a\}$



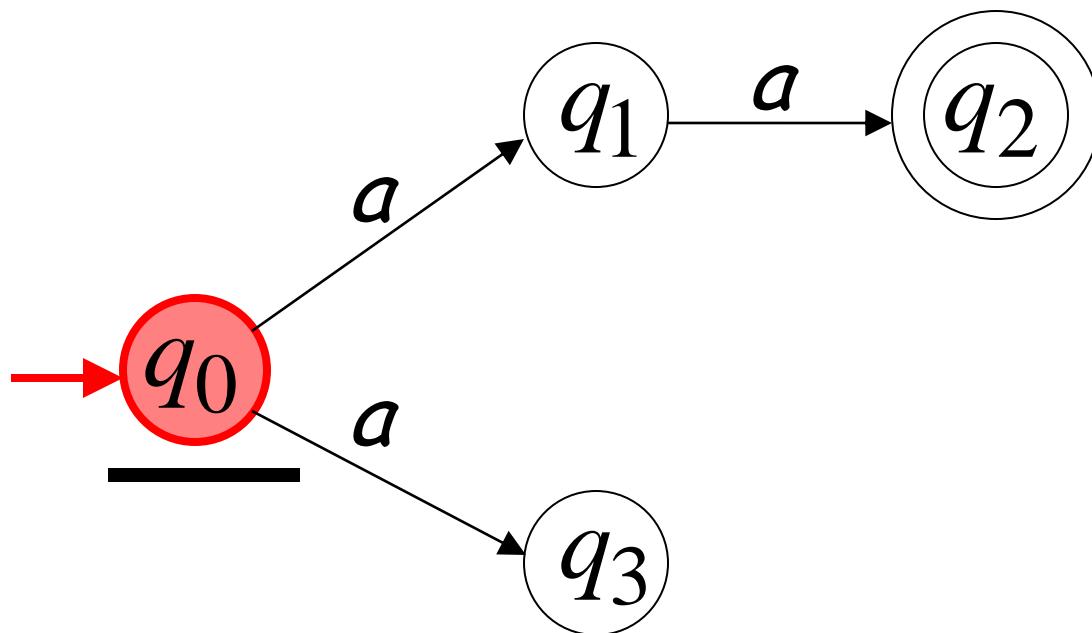
Alphabet =  $\{a\}$



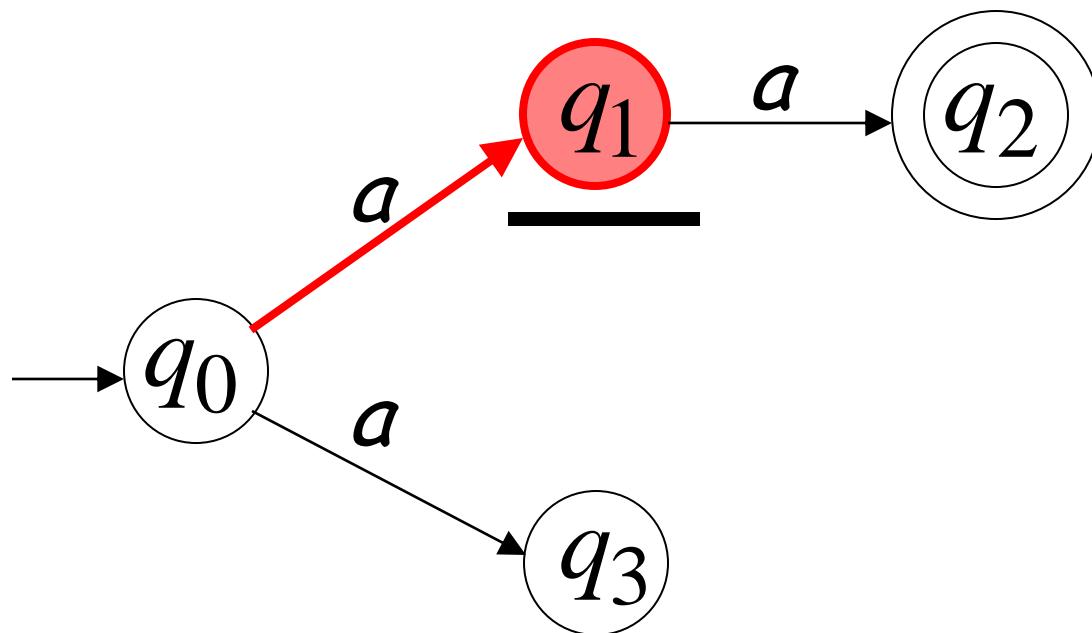
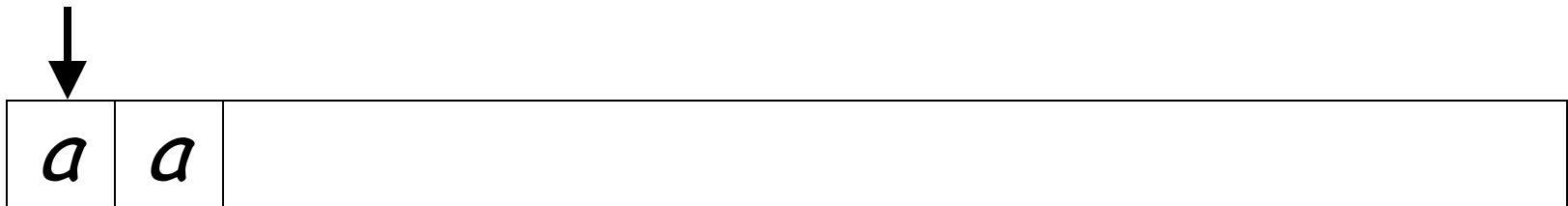
# First Choice



$a$	$a$	
-----	-----	--



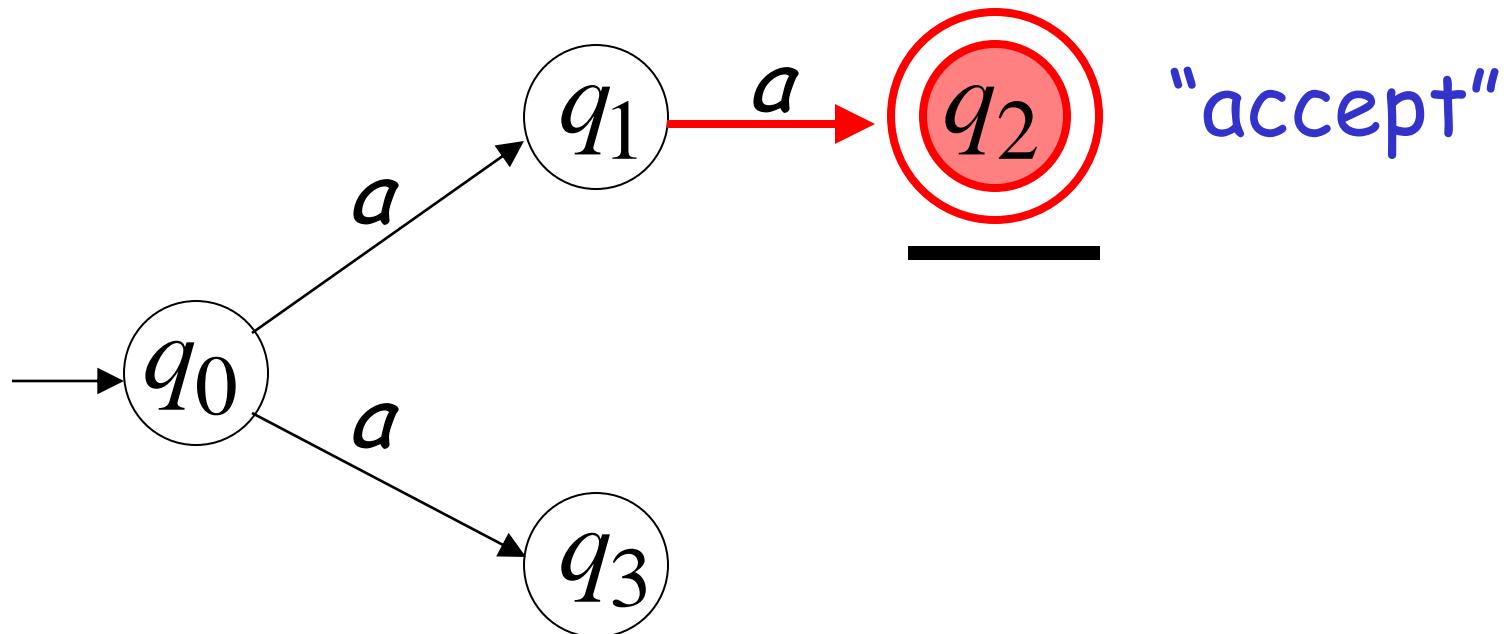
# First Choice



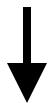
# First Choice



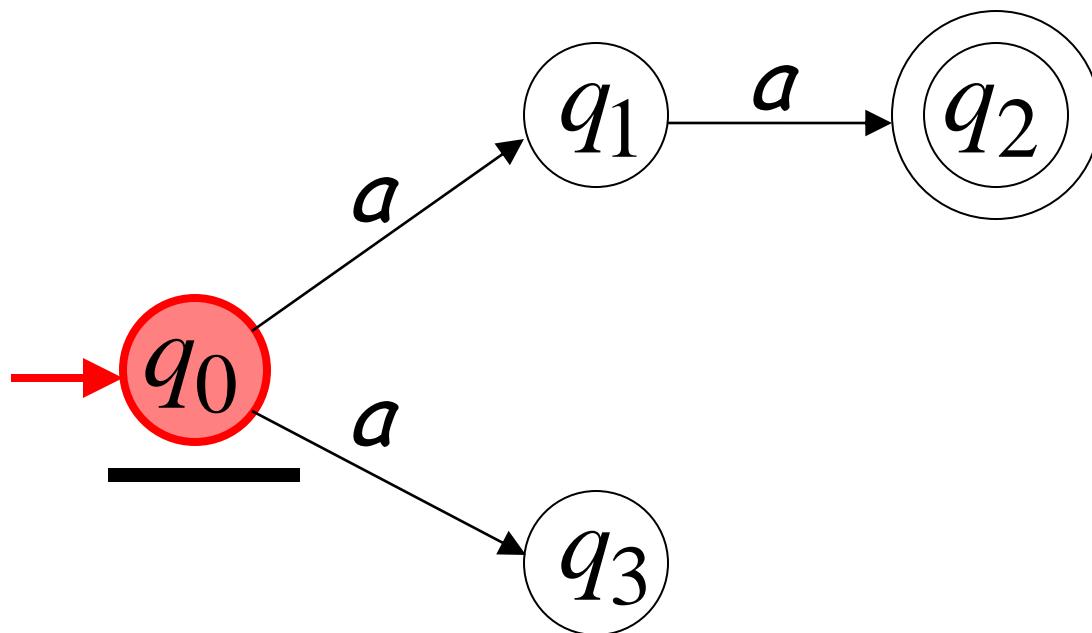
All input is consumed



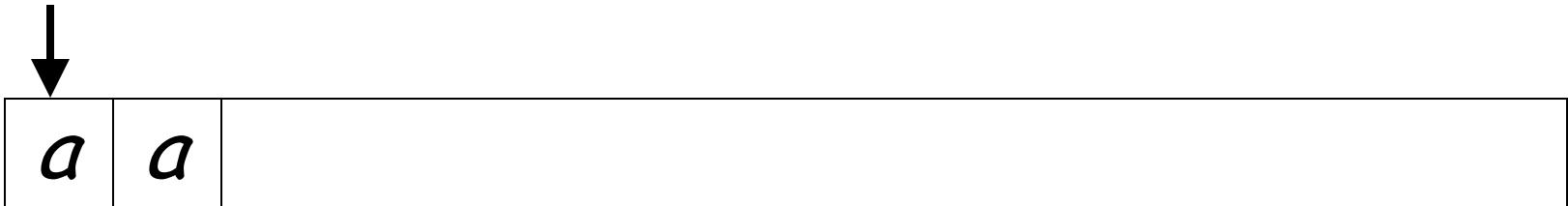
# Second Choice



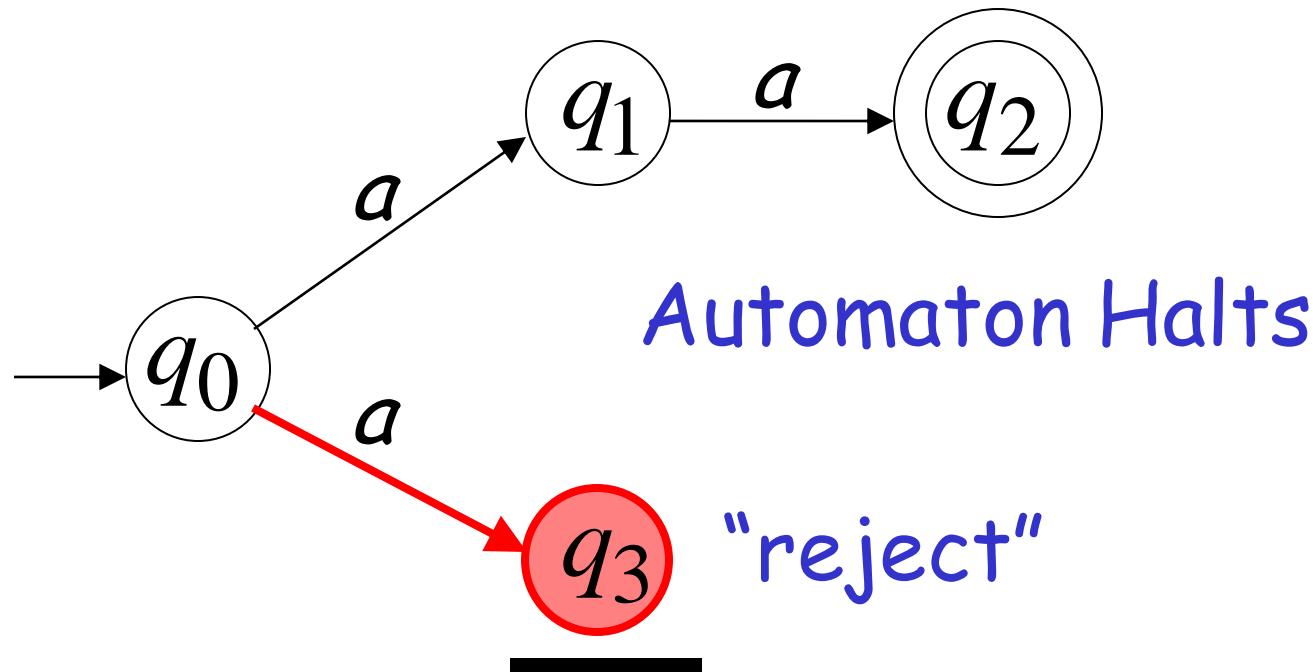
$a$	$a$	
-----	-----	--



# Second Choice



Input cannot be consumed

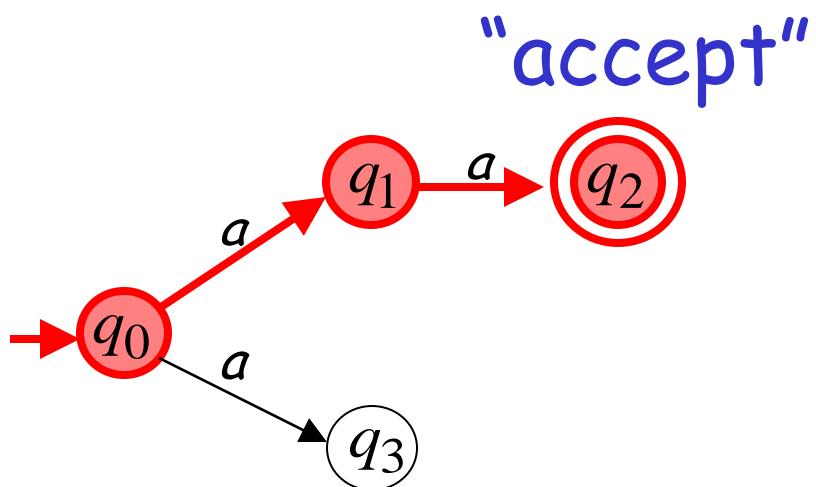


An NFA accepts a string:

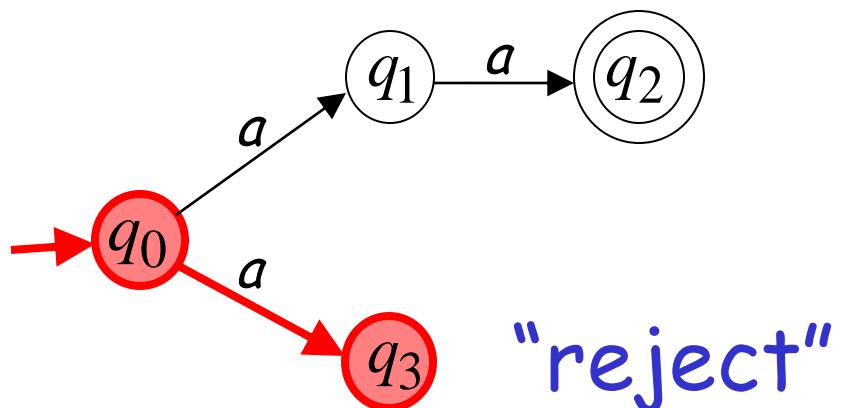
if there is a computation of the NFA  
that accepts the string

i.e., all the input string is processed and the  
automaton is in an accepting state

$aa$  is accepted by the NFA:



because this computation accepts  $aa$

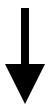


this computation is ignored

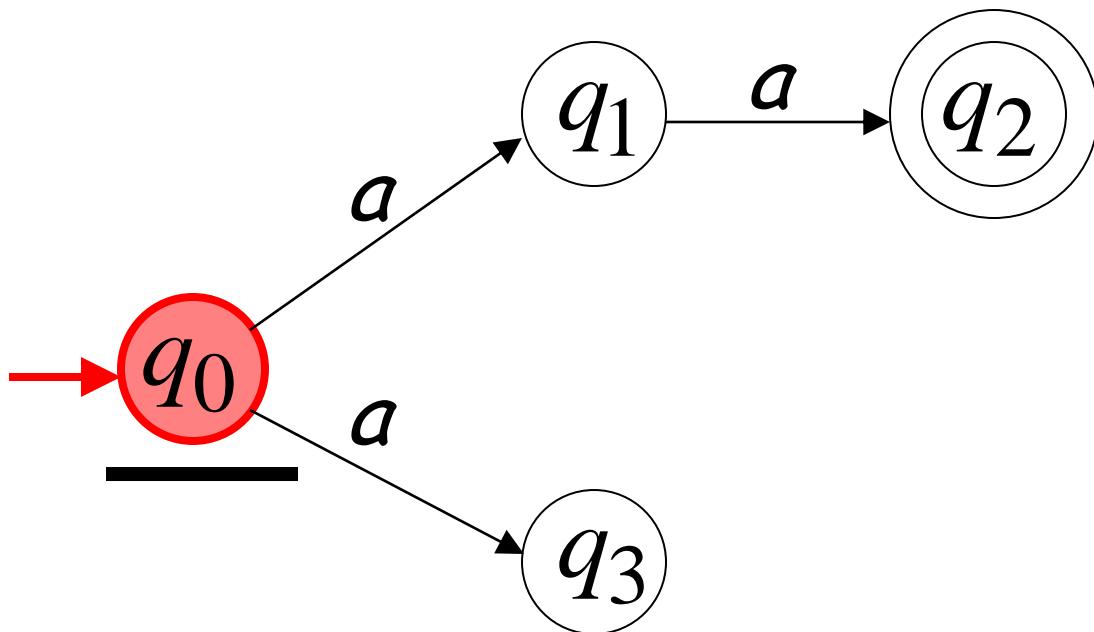
Absent No 27/6/19

2,11,16,25,27,28,34,37,38,42,43,53,54,  
59,65,69,71,72,74

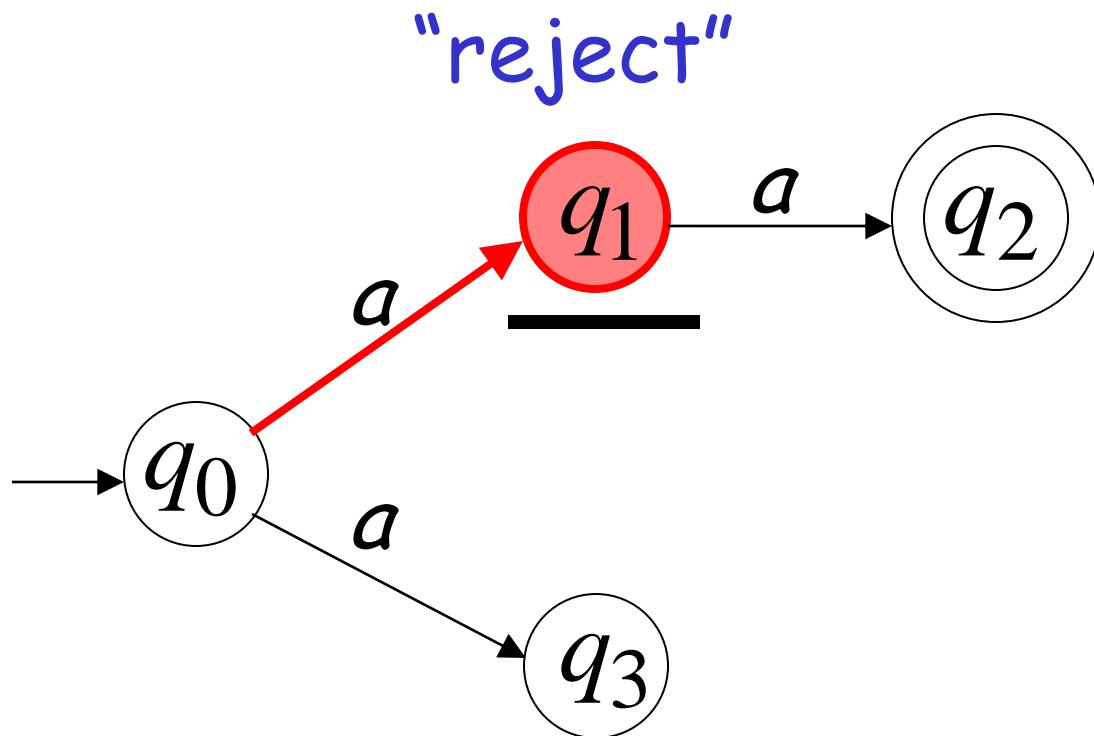
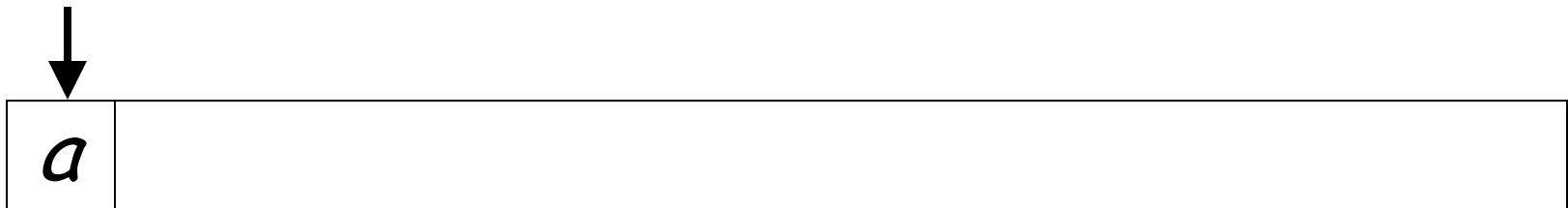
# Rejection example



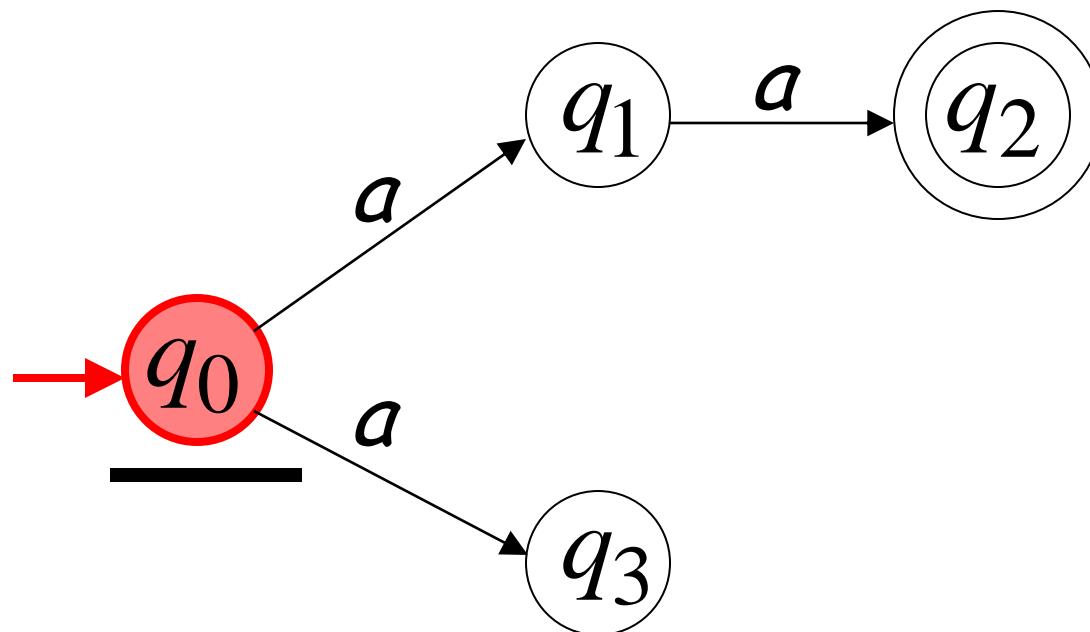
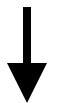
$a$	
-----	--



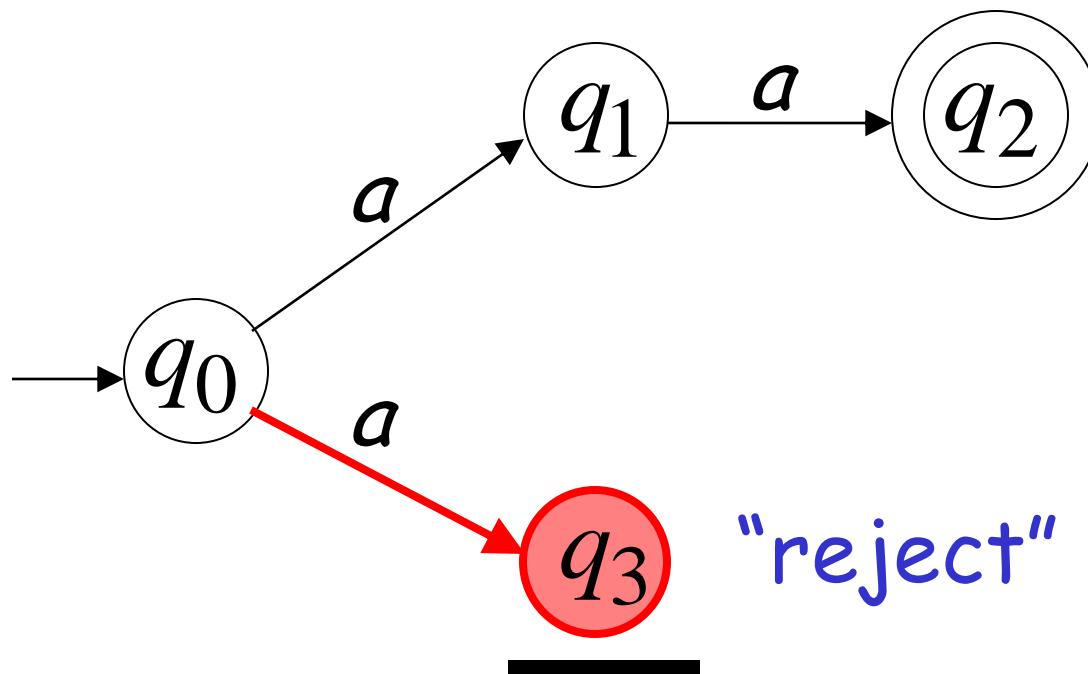
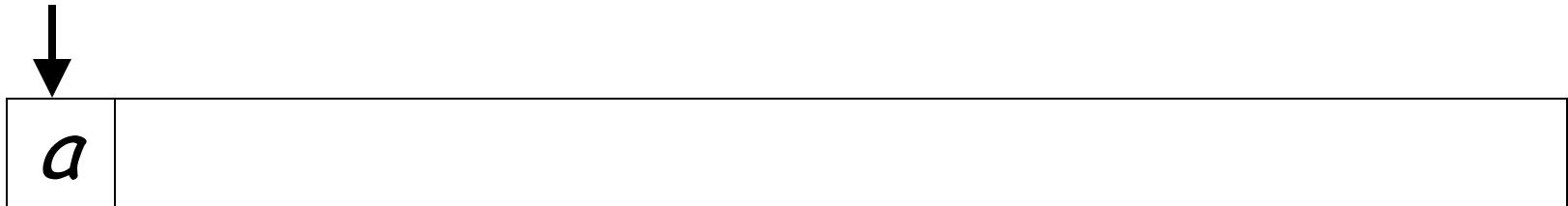
# First Choice



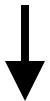
# Second Choice



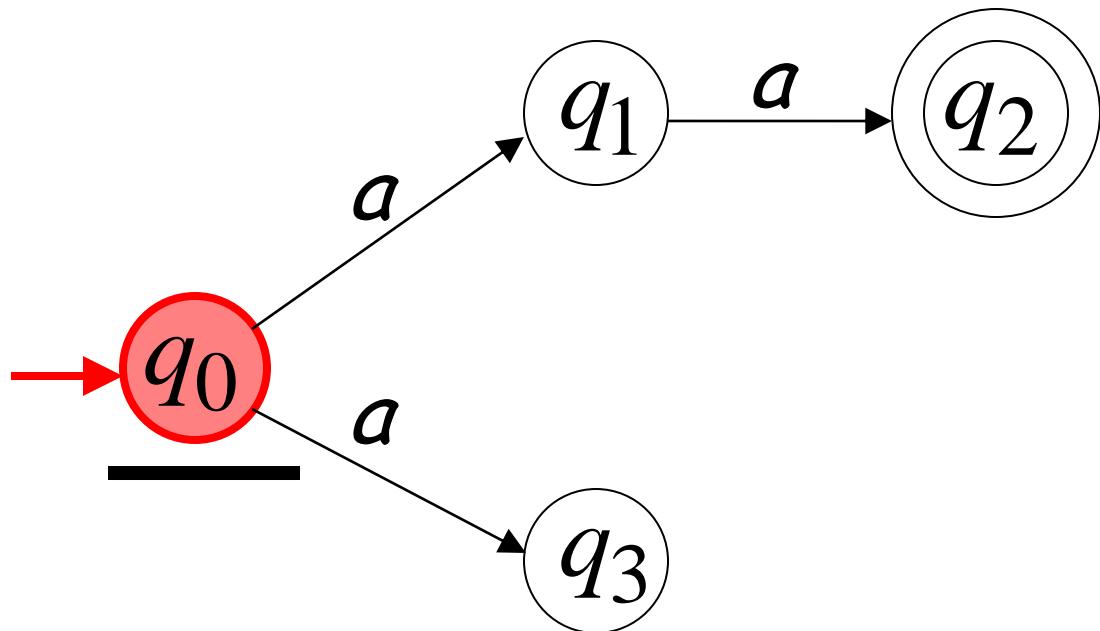
# Second Choice



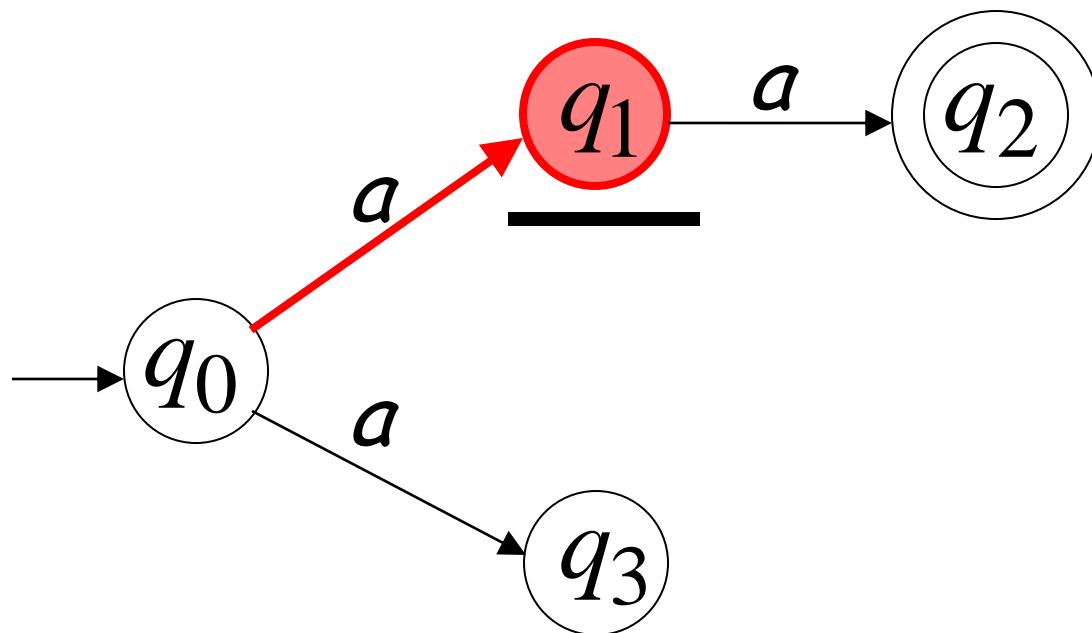
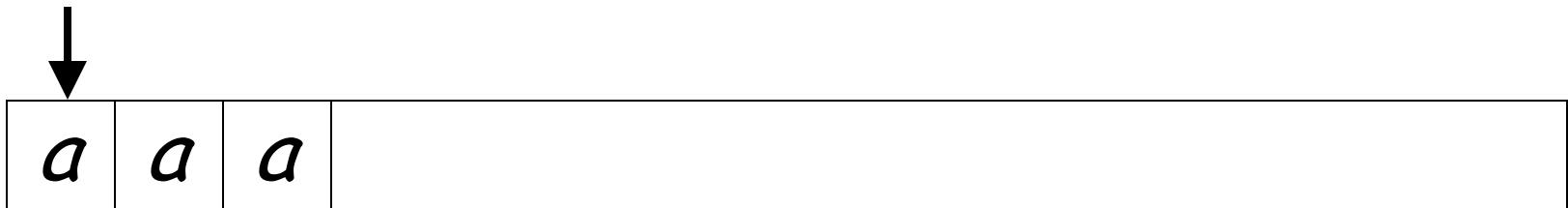
# Another Rejection example



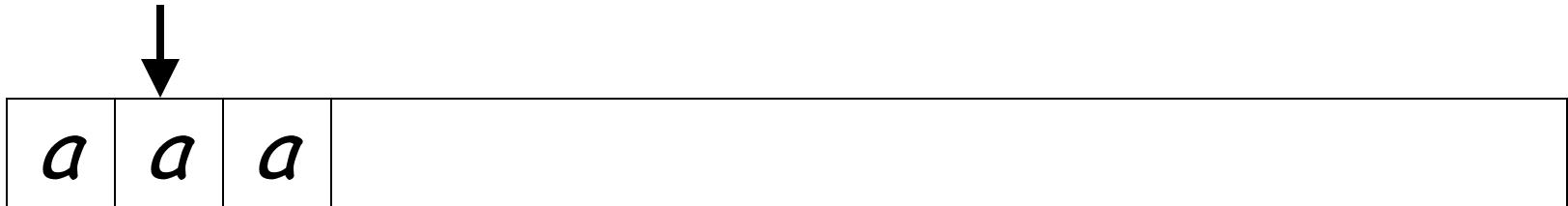
$a$	$a$	$a$	
-----	-----	-----	--



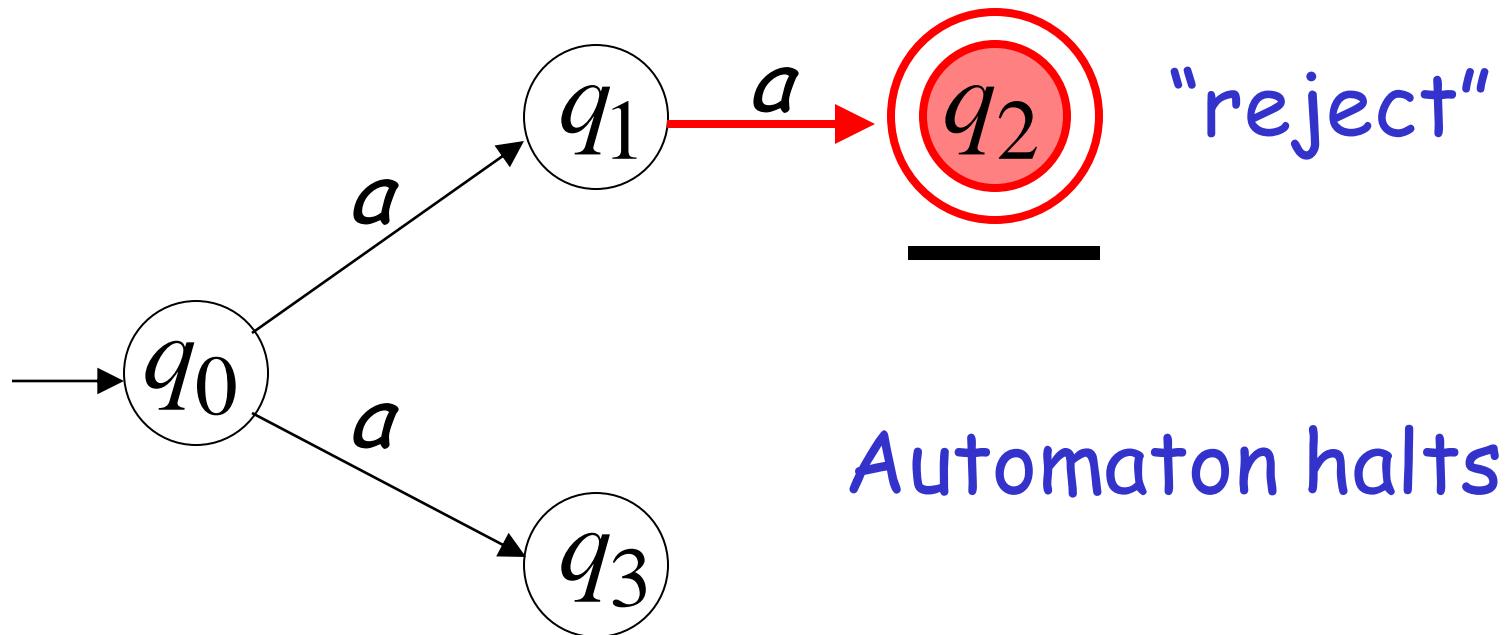
# First Choice



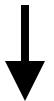
# First Choice



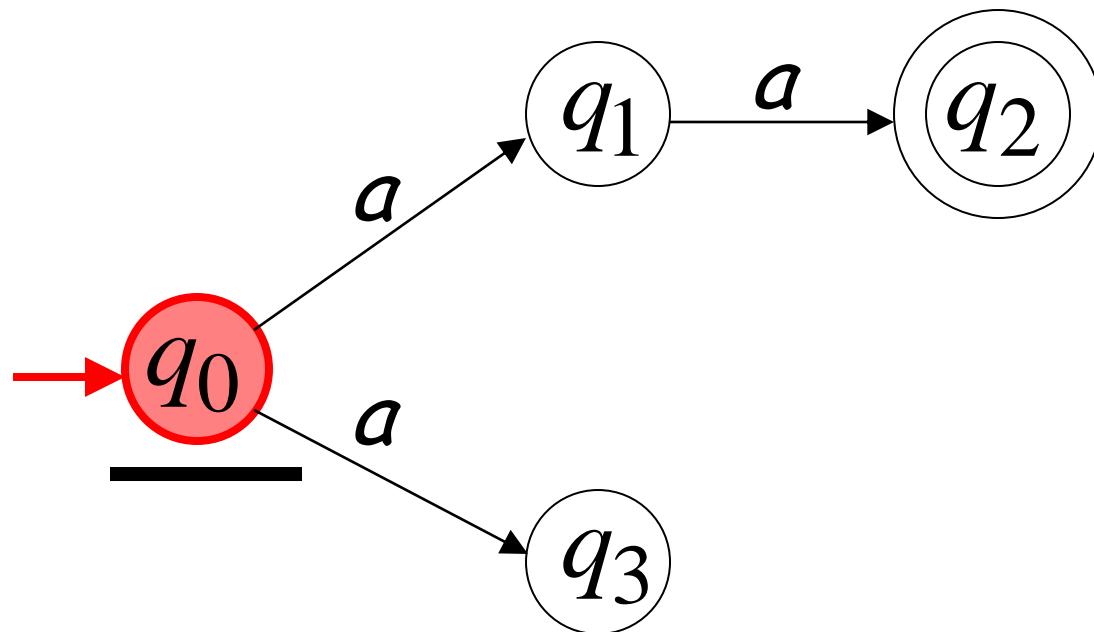
Input cannot be consumed



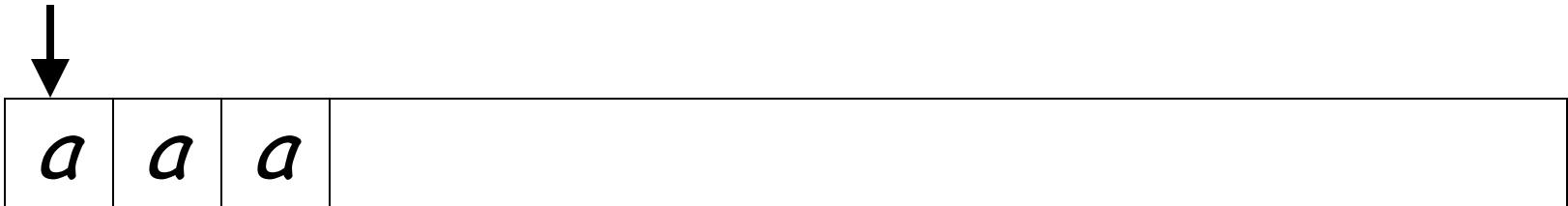
# Second Choice



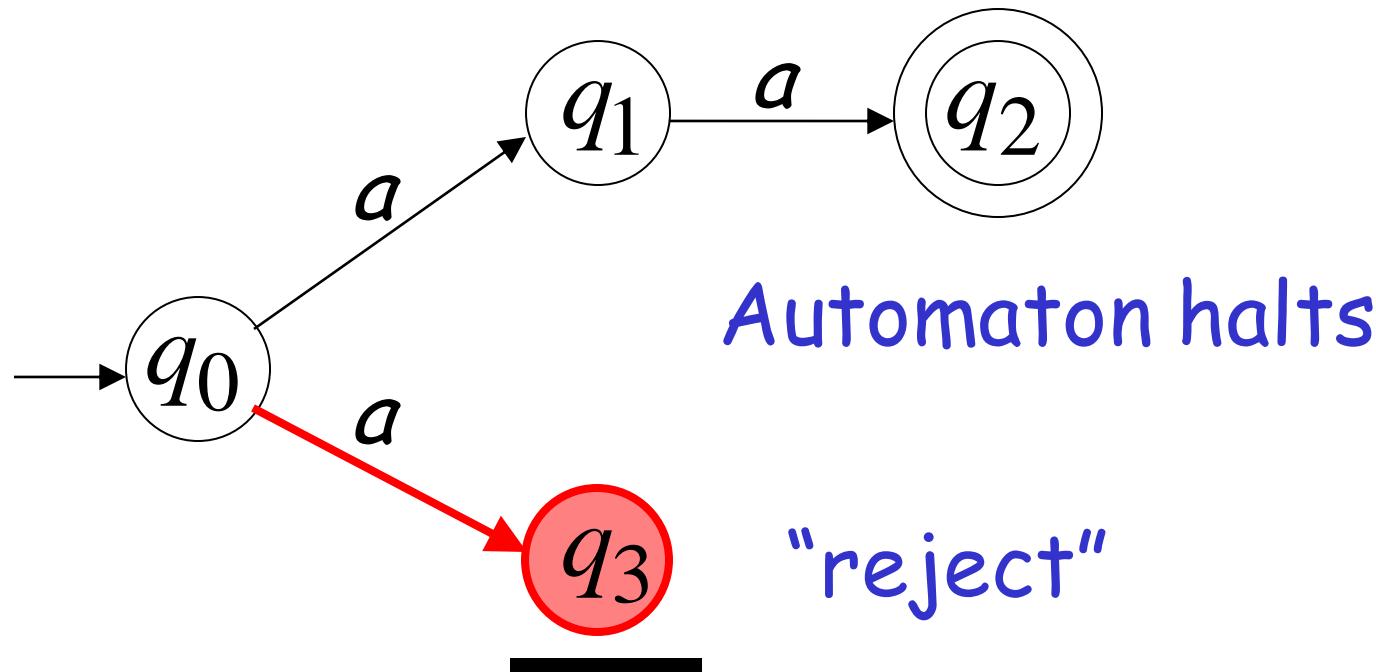
$a$	$a$	$a$	
-----	-----	-----	--



# Second Choice



Input cannot be consumed



An NFA rejects a string:

if there is no computation of the NFA  
that accepts the string.

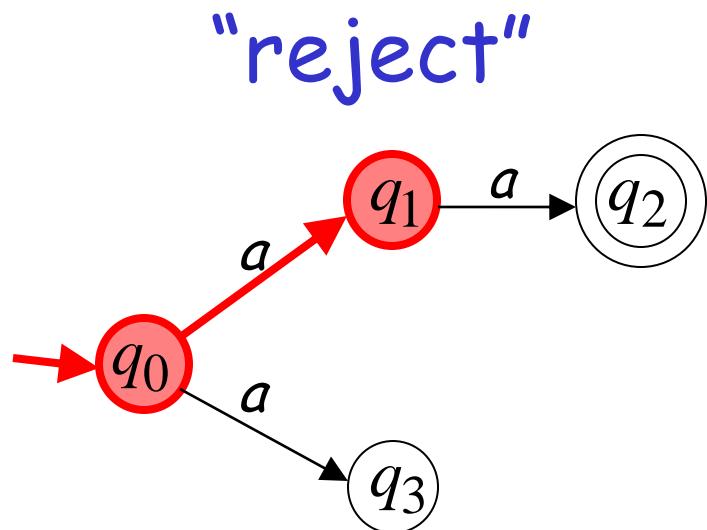
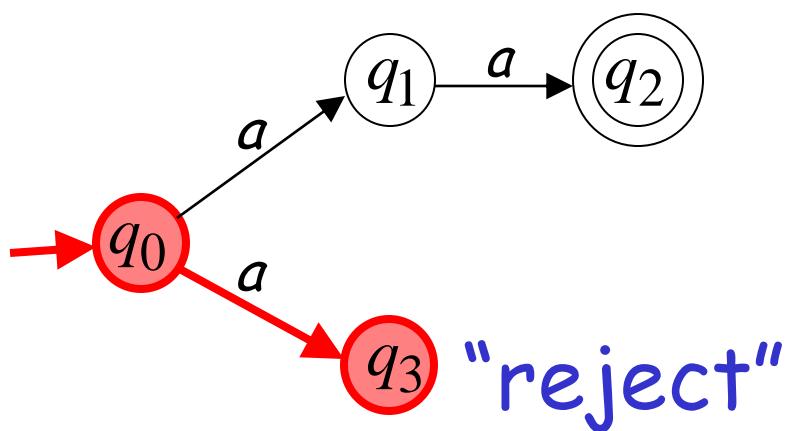
For each computation:

- All the input is consumed and the automaton is in a non final state

OR

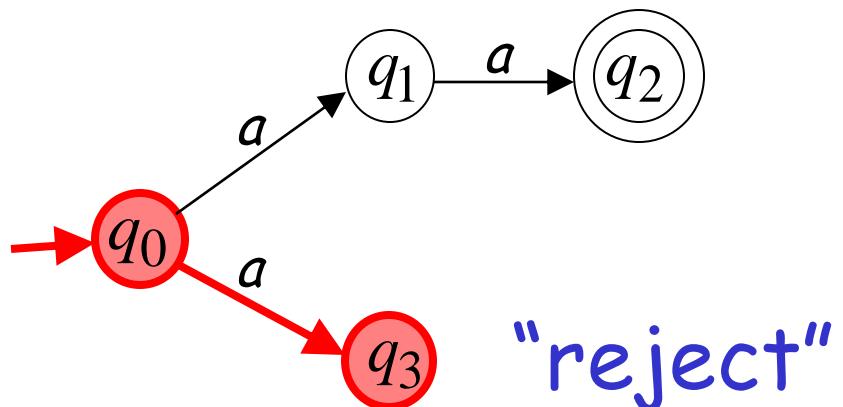
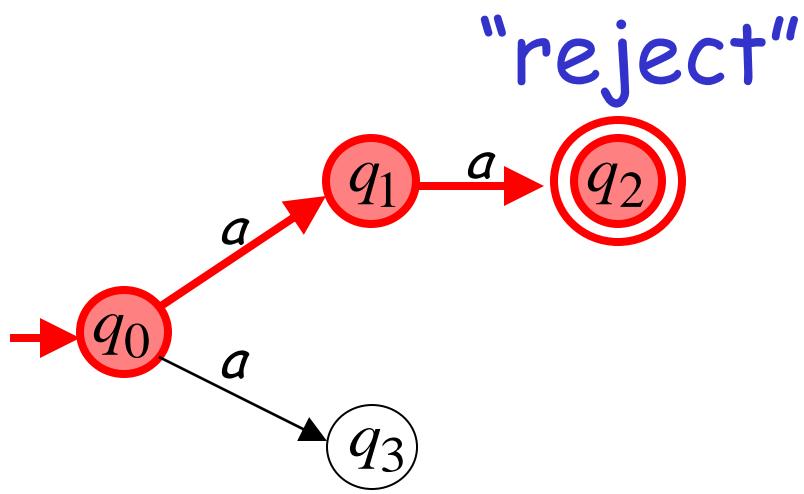
- The input cannot be consumed

a is rejected by the NFA:



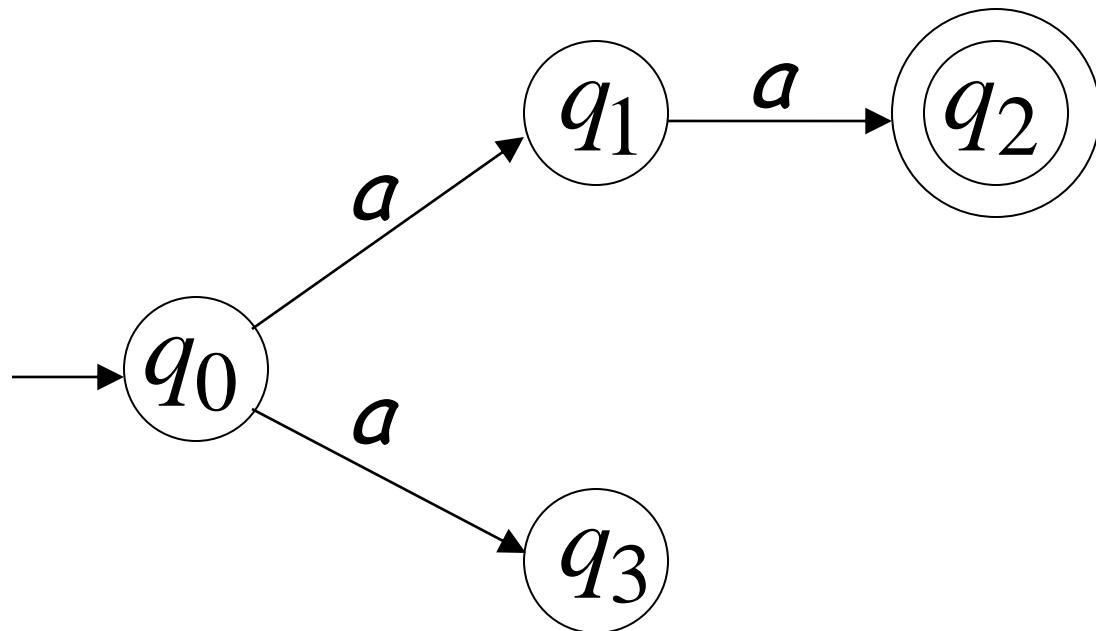
All possible computations lead to rejection

**aaa** is rejected by the NFA:

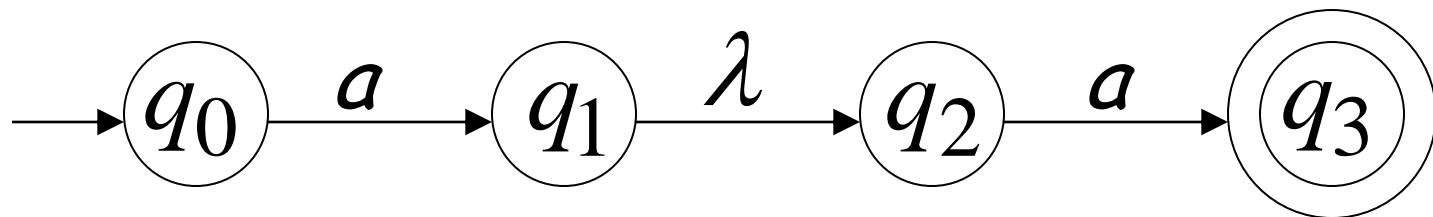


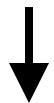
All possible computations lead to rejection

Language accepted:  $L = \{aa\}$

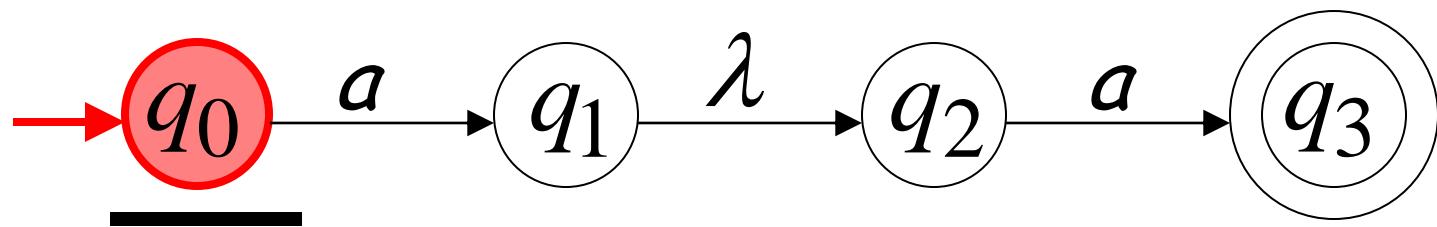


# Lambda Transitions



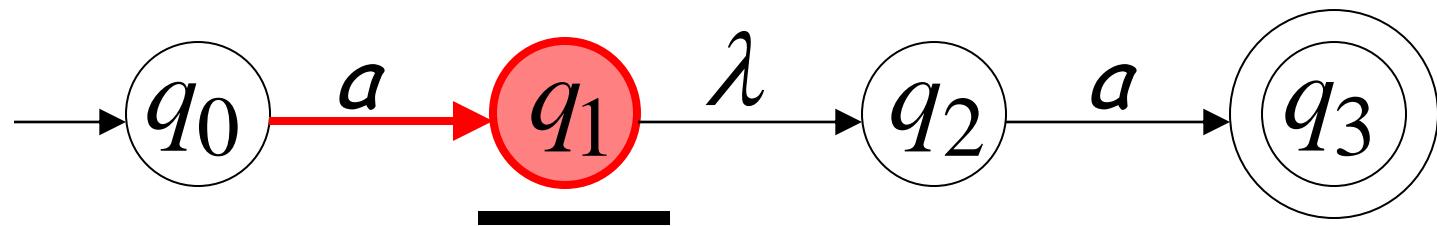


$a$	$a$	
-----	-----	--

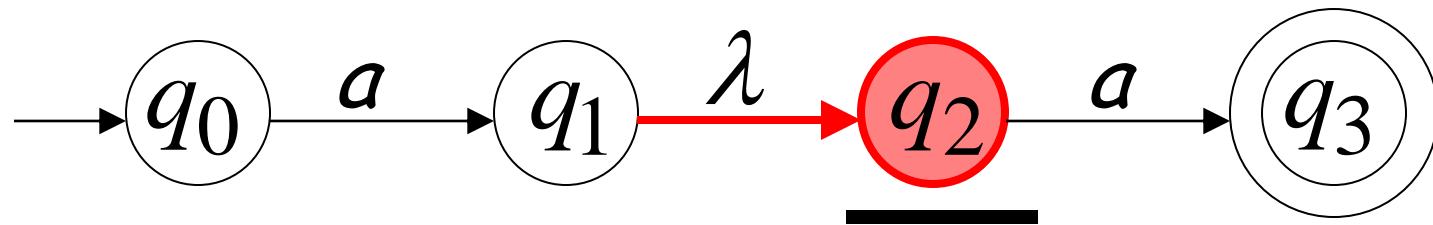
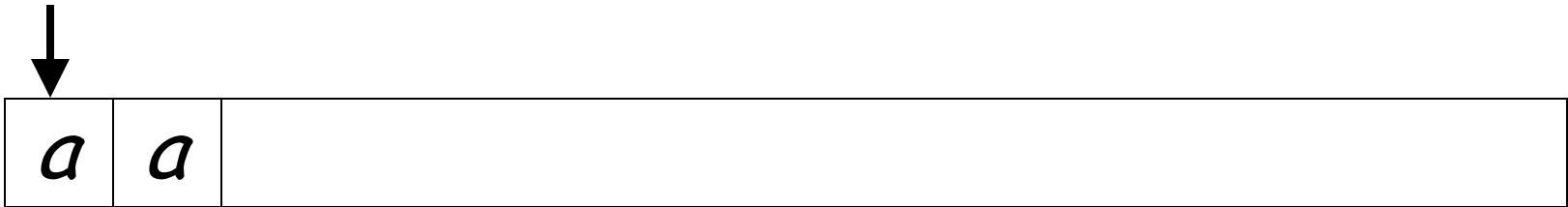


↓

$a$	$a$	
-----	-----	--



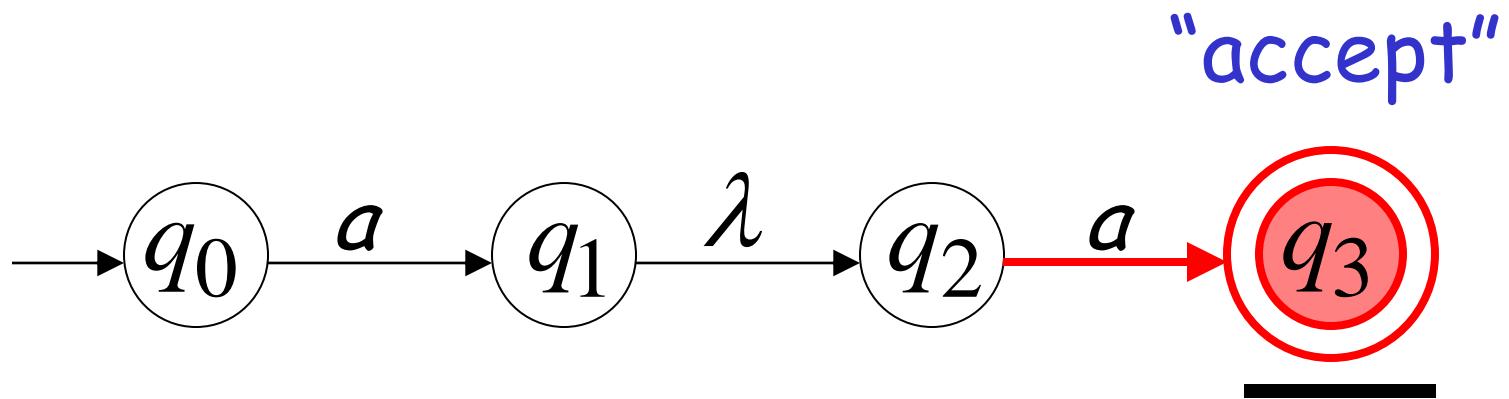
input tape head does not move



all input is consumed

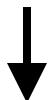


$a$	$a$	
-----	-----	--

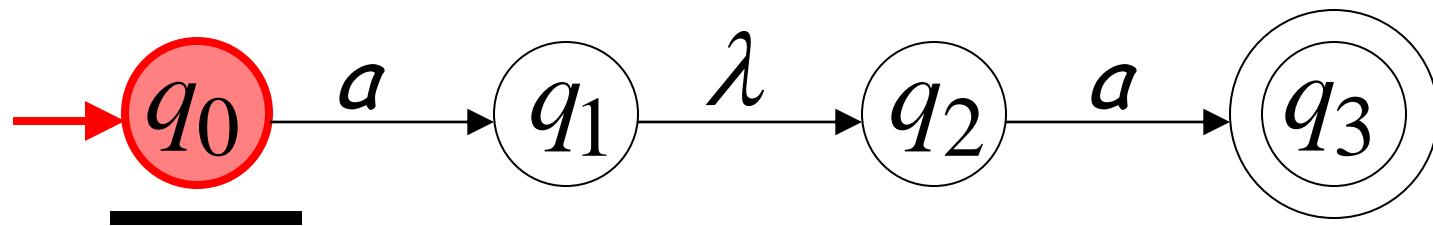


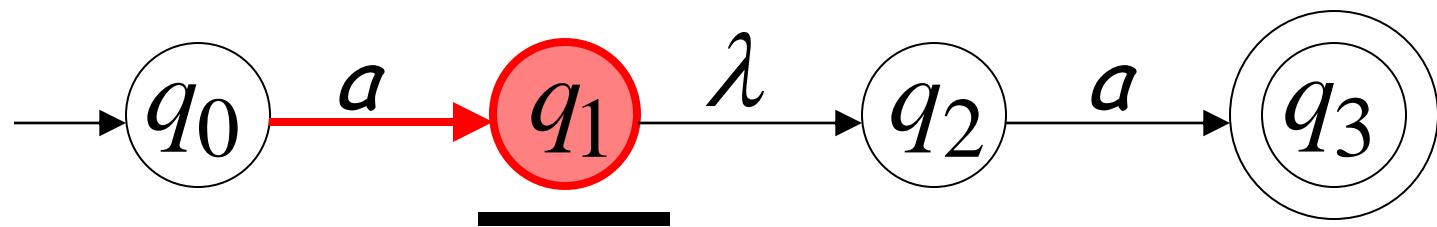
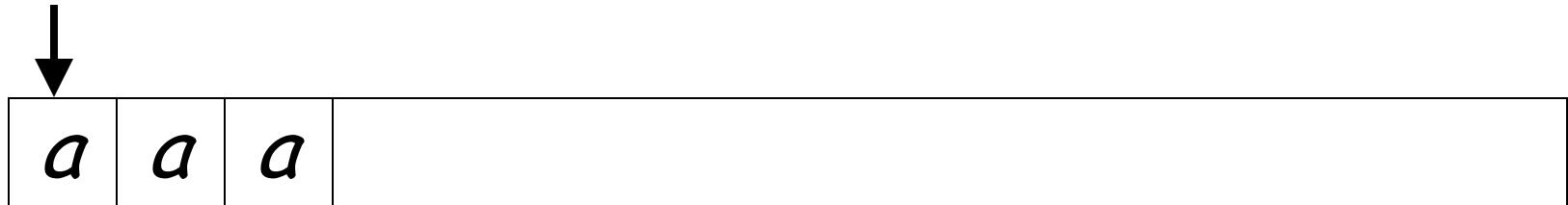
String  $aa$  is accepted

# Rejection Example

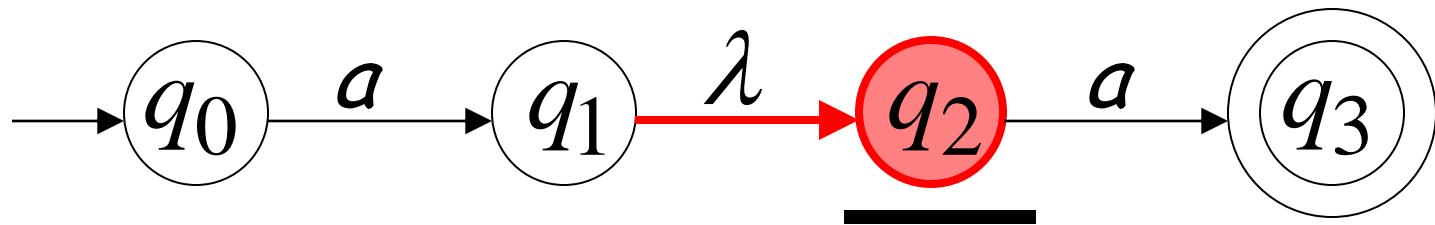
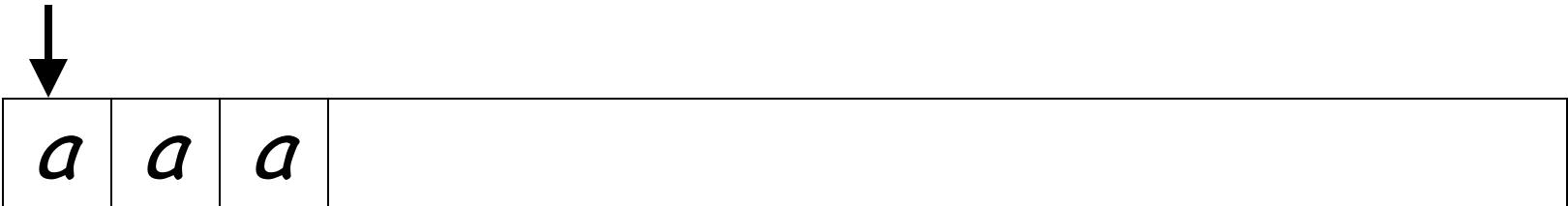


a	a	a	
---	---	---	--





(read head doesn't move)



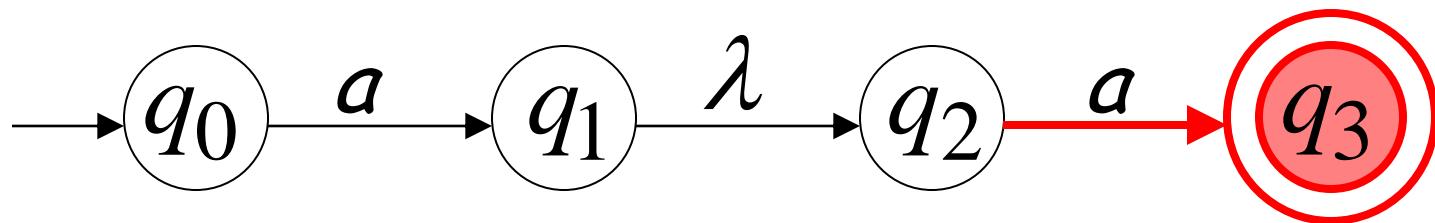
# Input cannot be consumed



a	a	a	
---	---	---	--

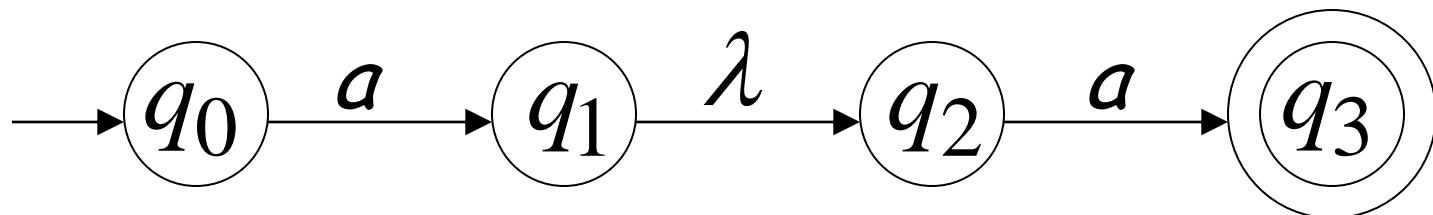
Automaton halts

"reject"

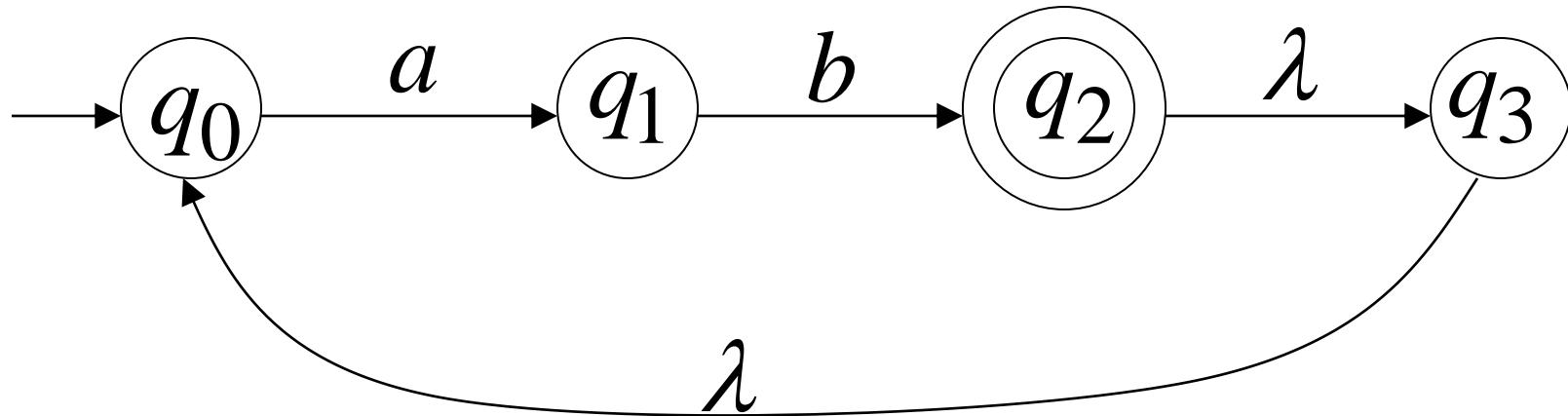


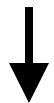
String aaa is rejected

Language accepted:  $L = \{aa\}$

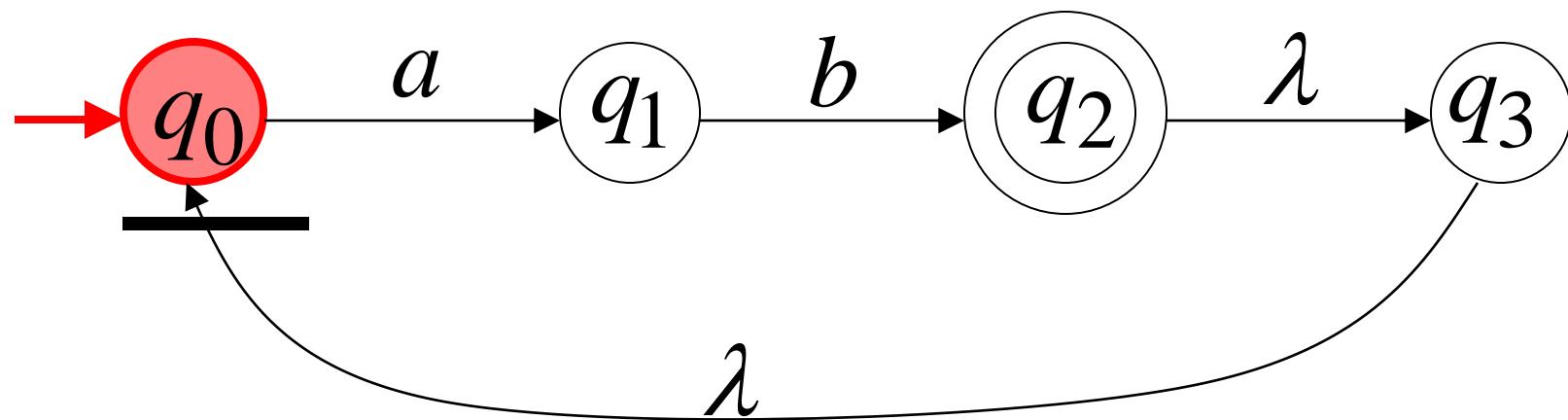


# Another NFA Example

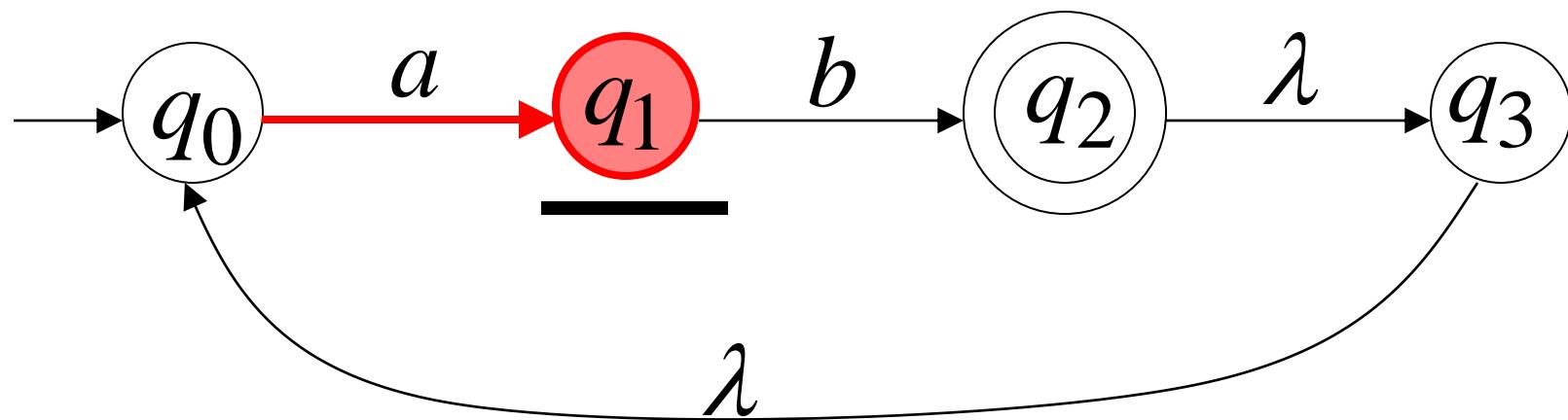




$a$	$b$	
-----	-----	--

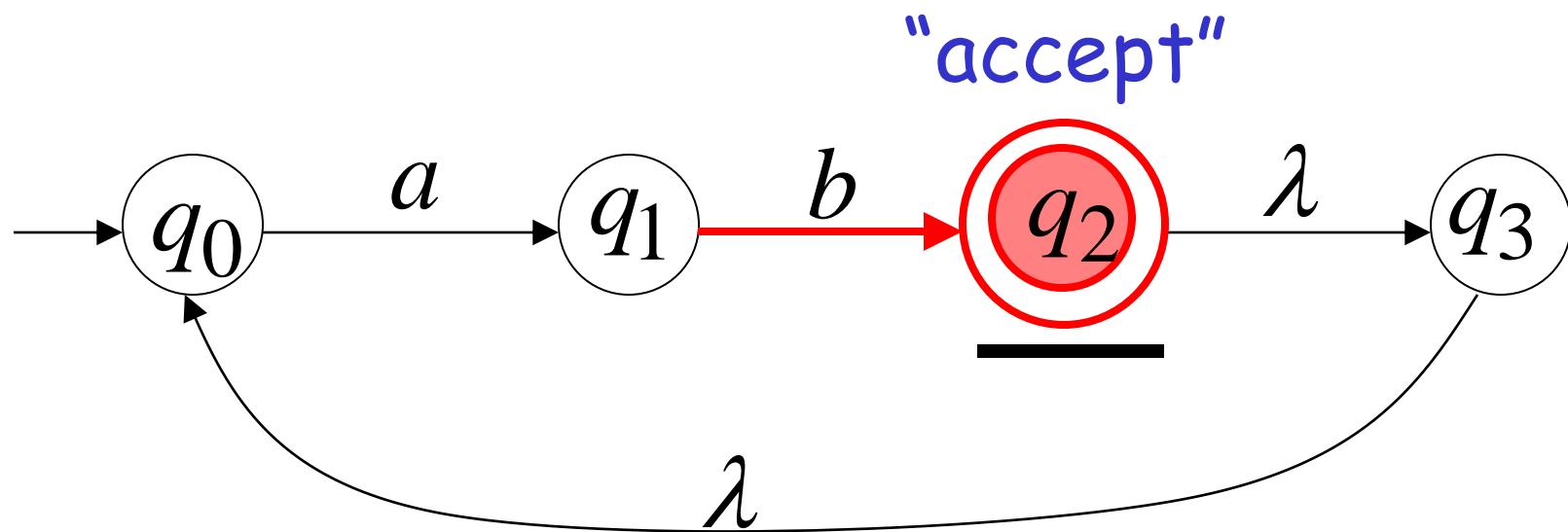


$a$	$b$	
-----	-----	--



↓

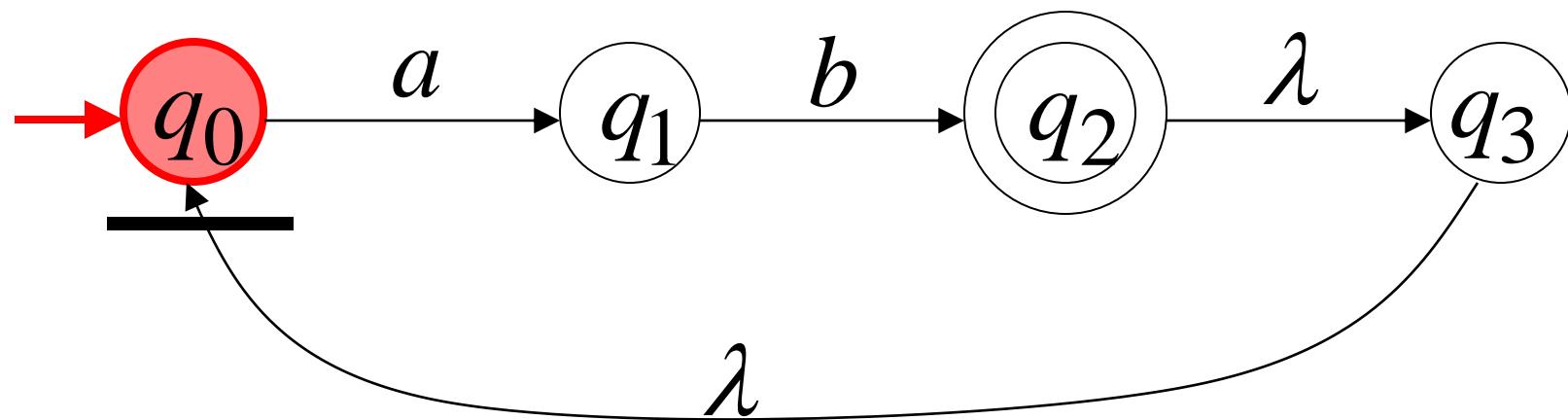
$a$	$b$	
-----	-----	--



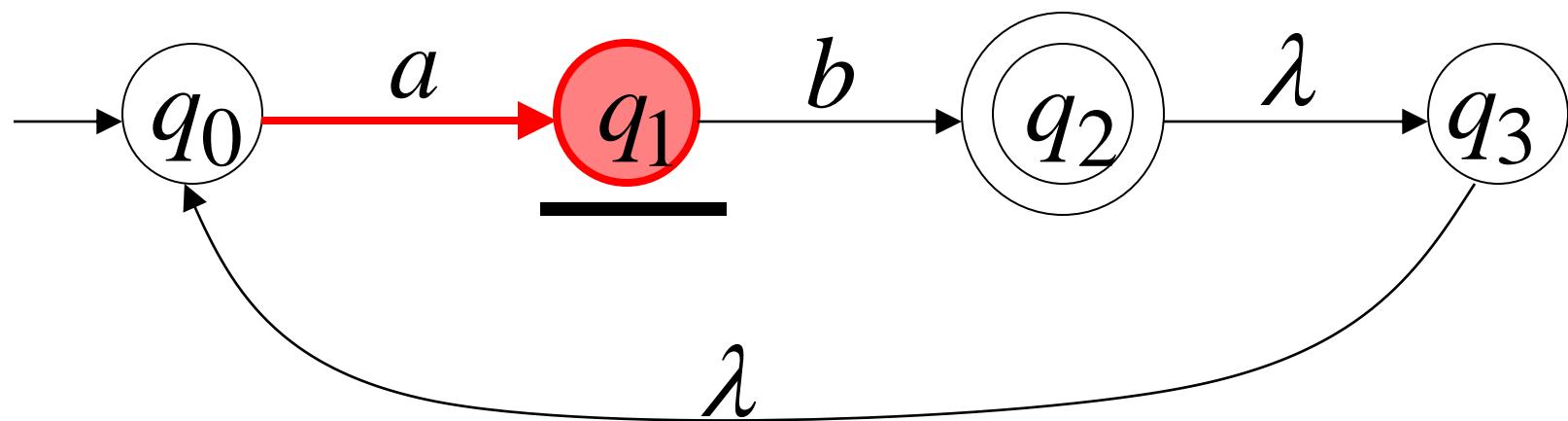
## Another String



$a$	$b$	$a$	$b$	
-----	-----	-----	-----	--

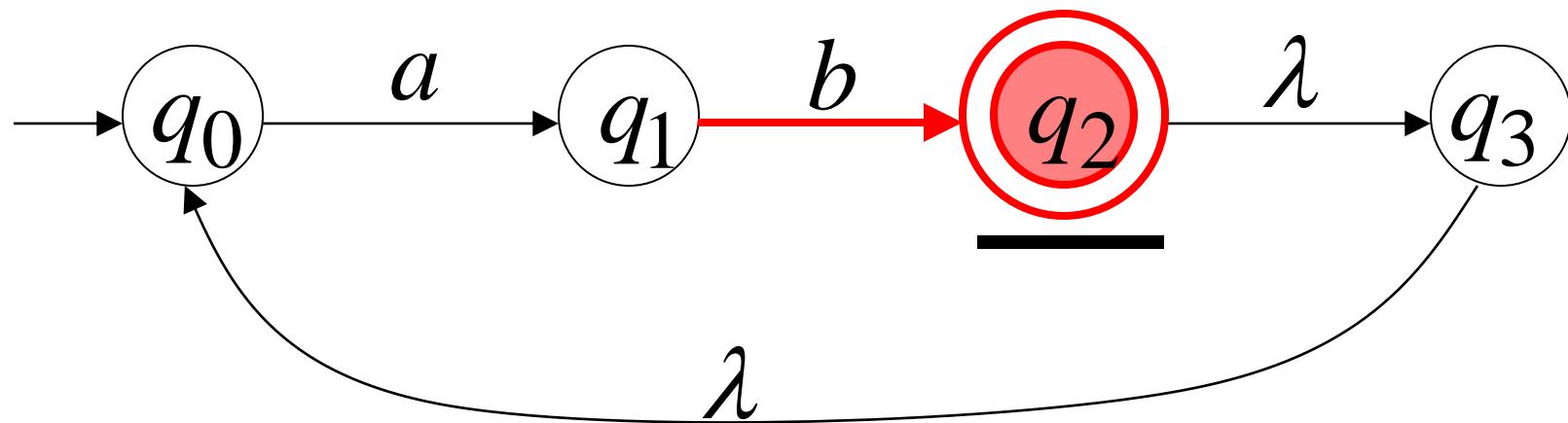


$a$	$b$	$a$	$b$	

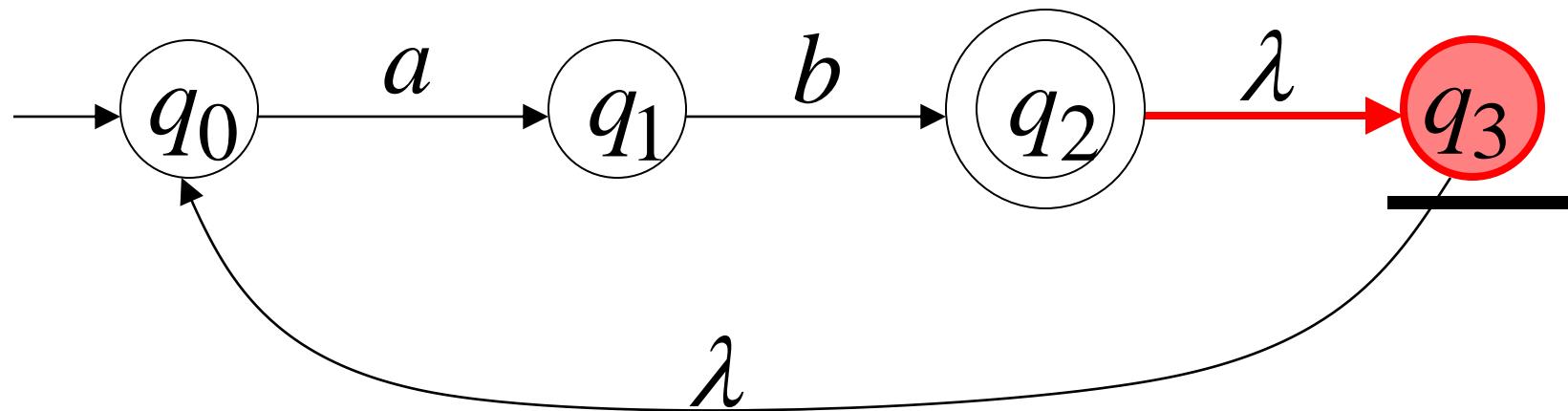


↓

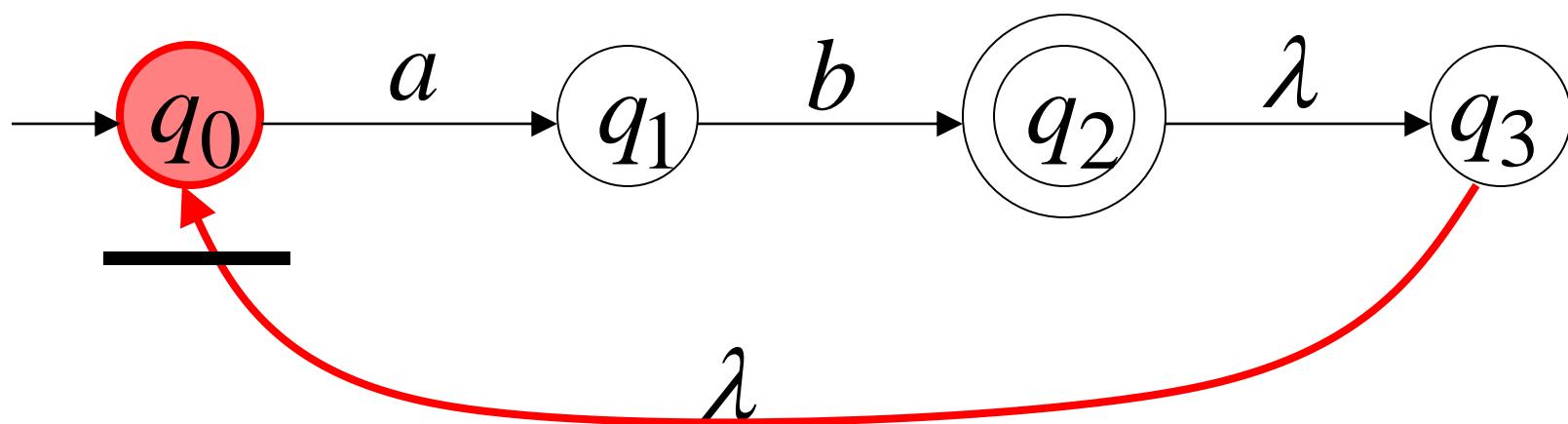
$a$	$b$	$a$	$b$	
-----	-----	-----	-----	--

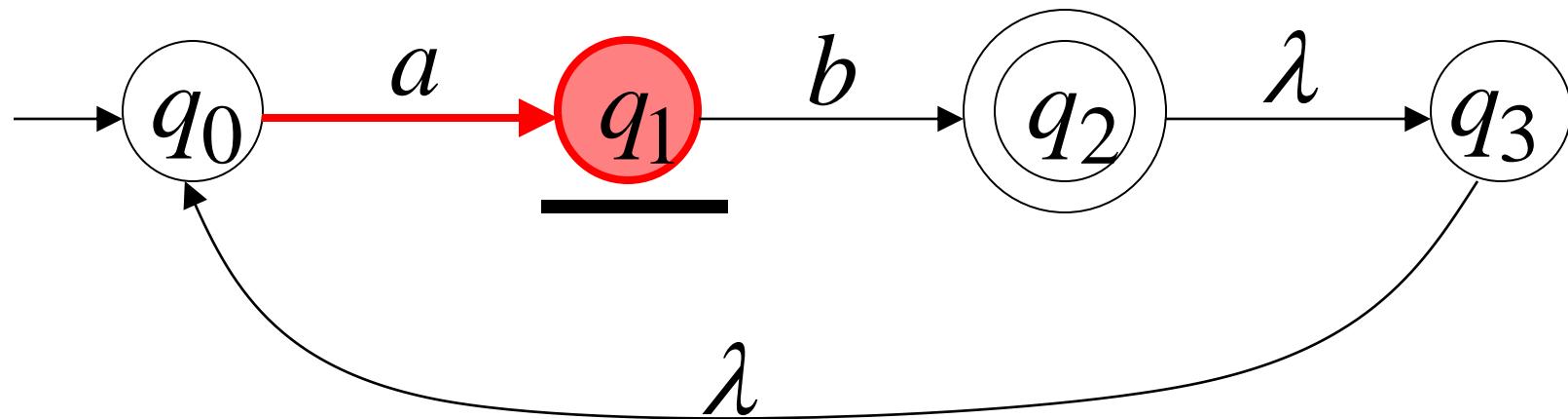
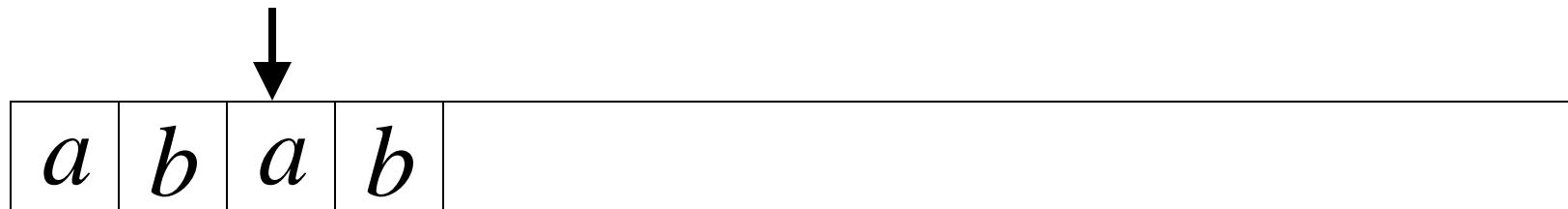


$a$	$b$	$a$	$b$	

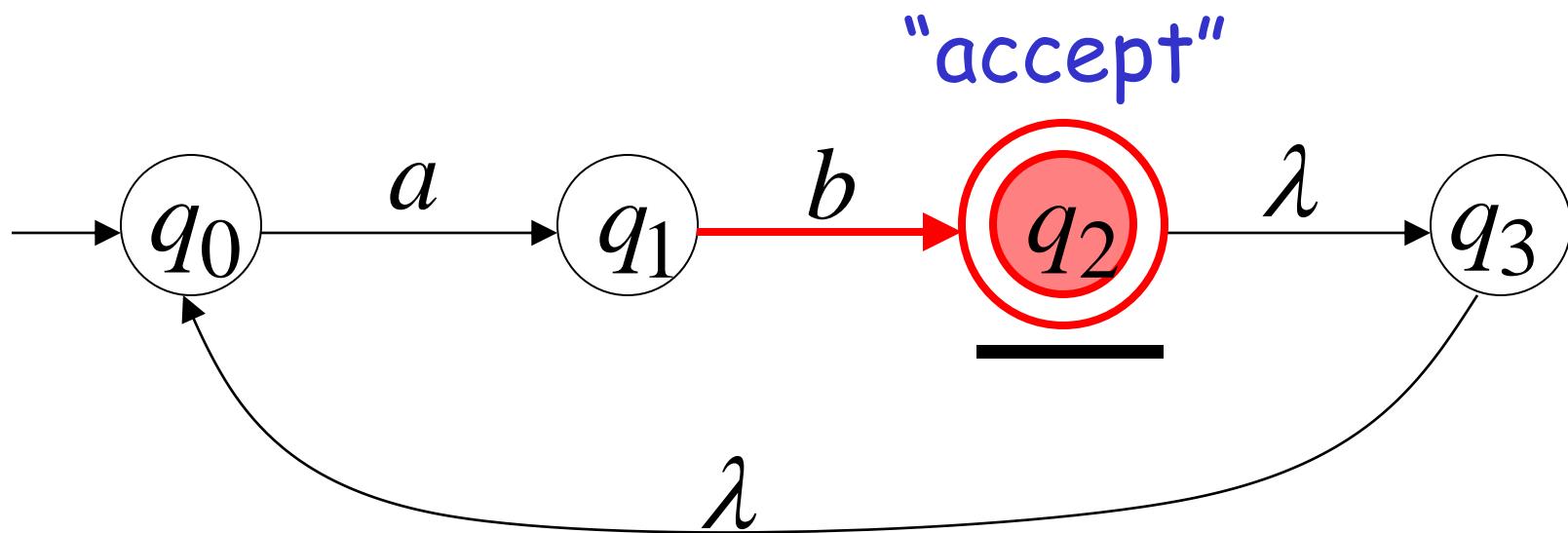


$a$	$b$	$a$	$b$	





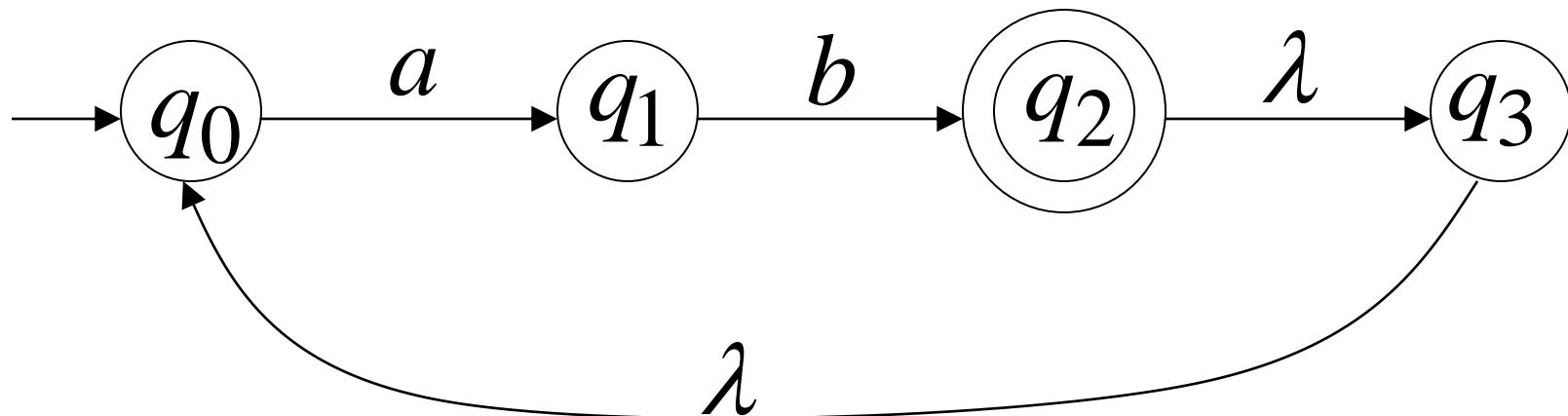
$a$	$b$	$a$	$b$	



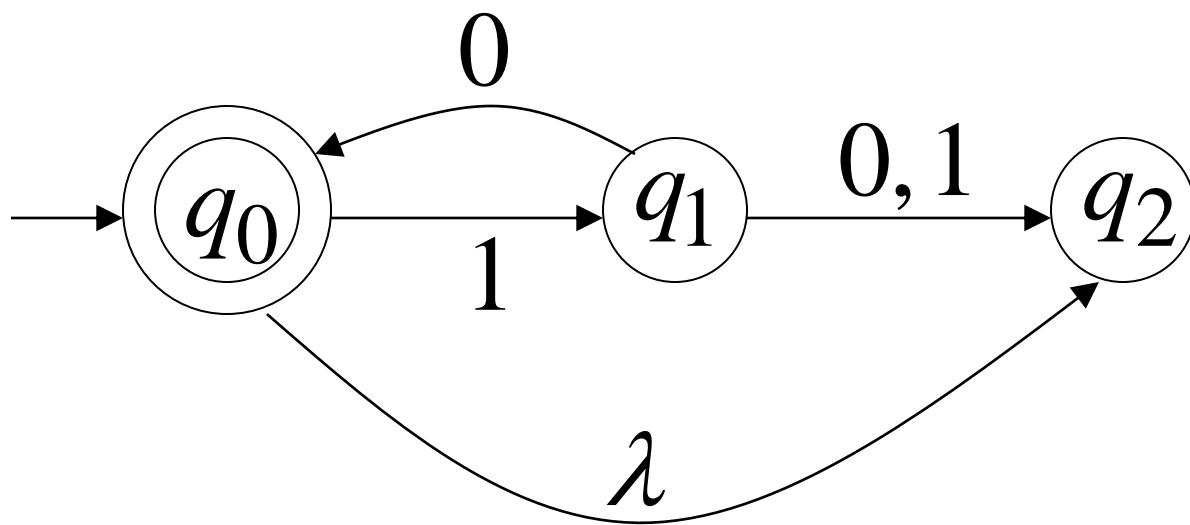
Language accepted

$$L = \{ab, abab, ababab, \dots\}$$

$$= \{ab\}^+$$

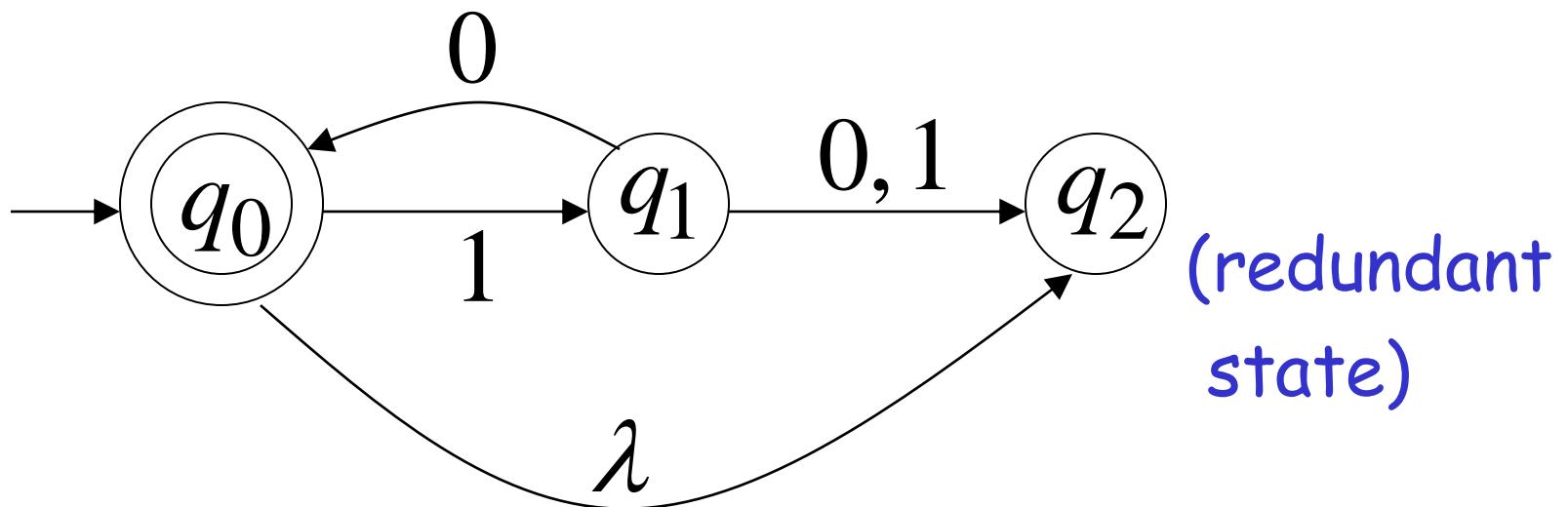


# Another NFA Example



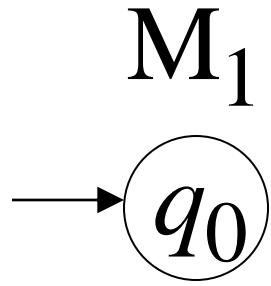
Language accepted

$$\begin{aligned}L(M) &= \{\lambda, 10, 1010, 101010, \dots\} \\&= \{10\}^*\end{aligned}$$

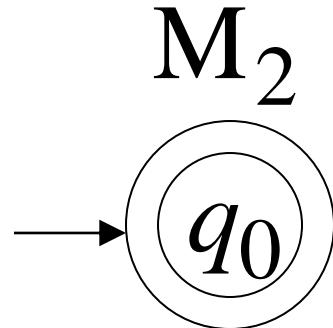


## Remarks:

- The  $\lambda$  symbol never appears on the input tape
- Simple automata:

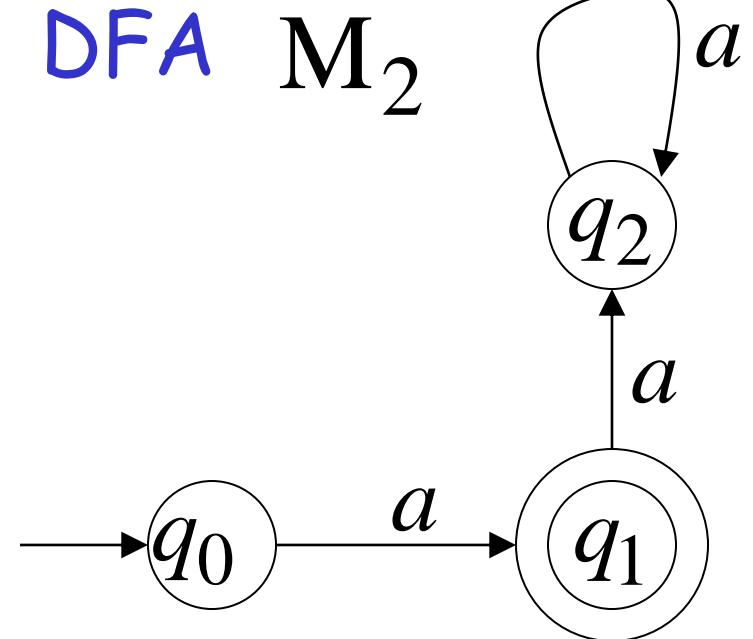
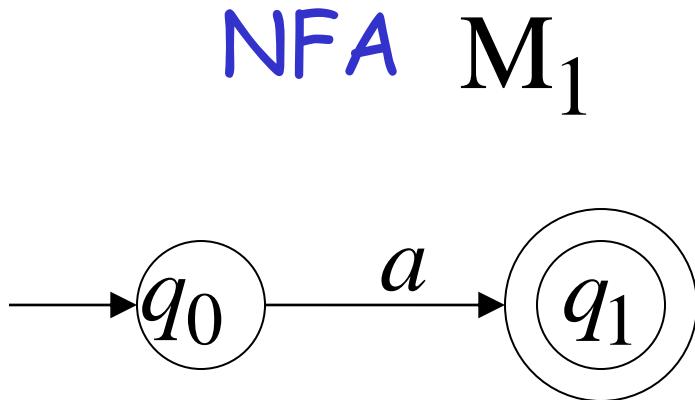


$$L(M_1) = \{ \}$$



$$L(M_2) = \{ \lambda \}$$

- NFAs are interesting because we can express languages easier than DFAs



$$L(M_1) = \{a\}$$

$$L(M_2) = \{a\}$$

# Formal Definition of NFAs

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

$Q$ : Set of states, i.e.  $\{q_0, q_1, q_2\}$

$\Sigma$ : Input alphabet, i.e.  $\{a, b\}$        $\lambda \notin \Sigma$

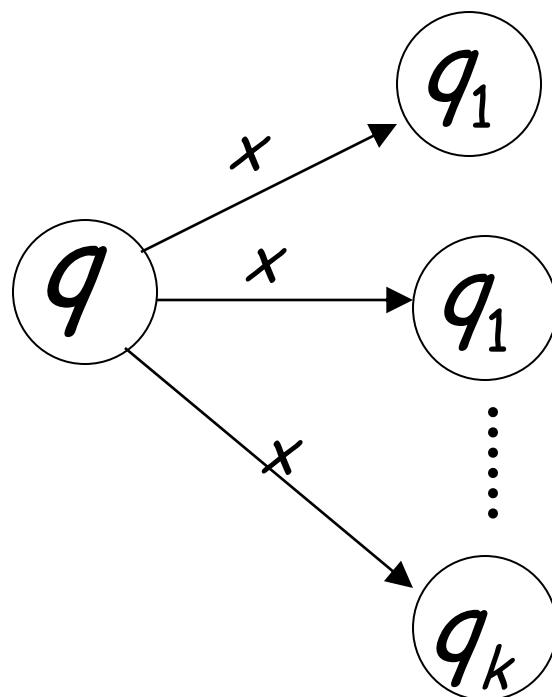
$\delta$ : Transition function

$q_0$ : Initial state

$F$ : Accepting states

# Transition Function $\delta$

$$\delta(q, x) = \{q_1, q_2, \dots, q_k\}$$



resulting states with  
following **one** transition  
with symbol  $x$

# Formal Notation - Epsilon Transition

- Transition function  $\delta$  is now a function that takes as arguments:
  - A state in  $Q$  and
  - A member of  $\Sigma \cup \{\epsilon\}$ ; that is, an input symbol or the symbol  $\epsilon$ . We require that  $\epsilon$  not be a symbol of the alphabet  $\Sigma$  to avoid any confusion.

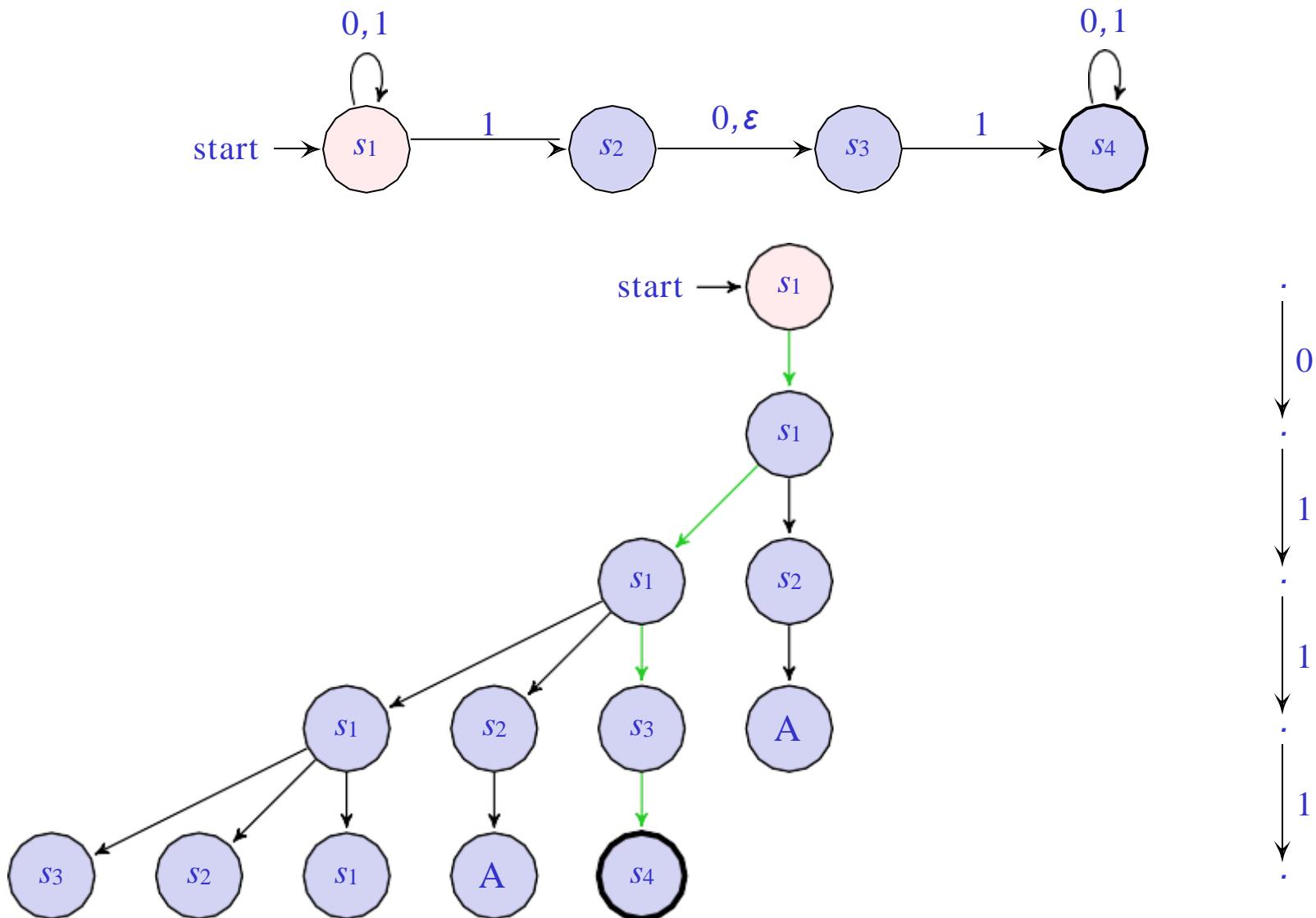
# NFA's With $\epsilon$ -Transitions

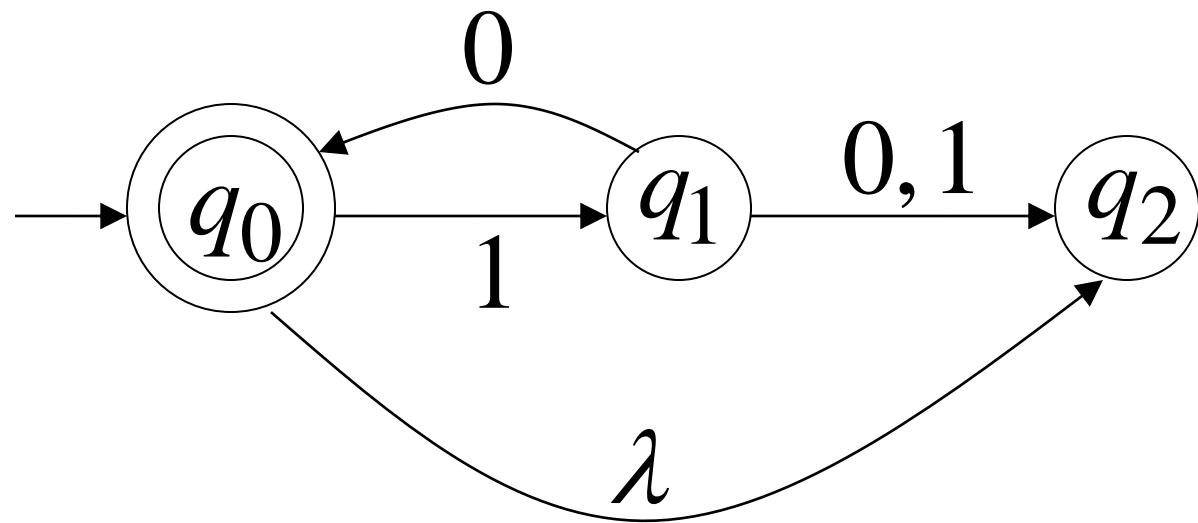
- We can allow state-to-state transitions on  $\epsilon$  input.
- These transitions are done spontaneously, without looking at the input string.
- A convenience at times, but still only regular languages are accepted.

# Corollary

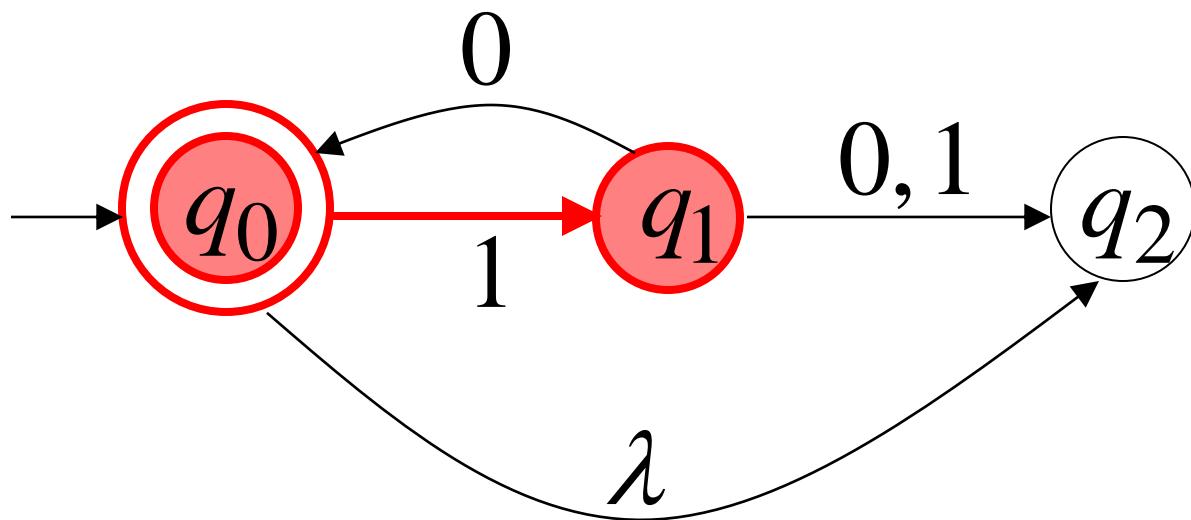
- A language is regular if and only if some nondeterministic finite automaton recognizes it
- A language is regular if and only if some deterministic finite automaton recognizes it

# Computation of an NFA

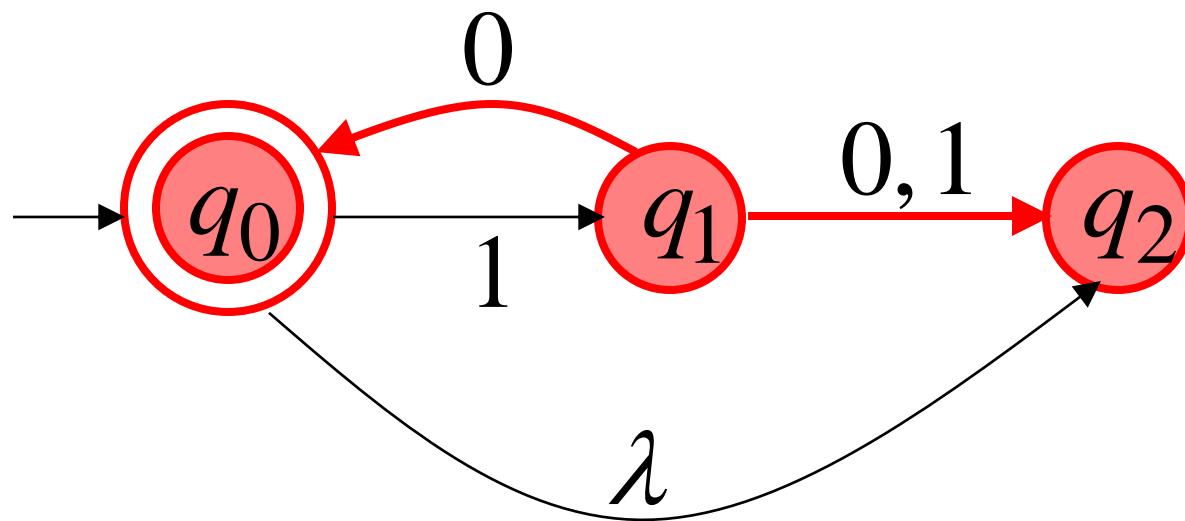




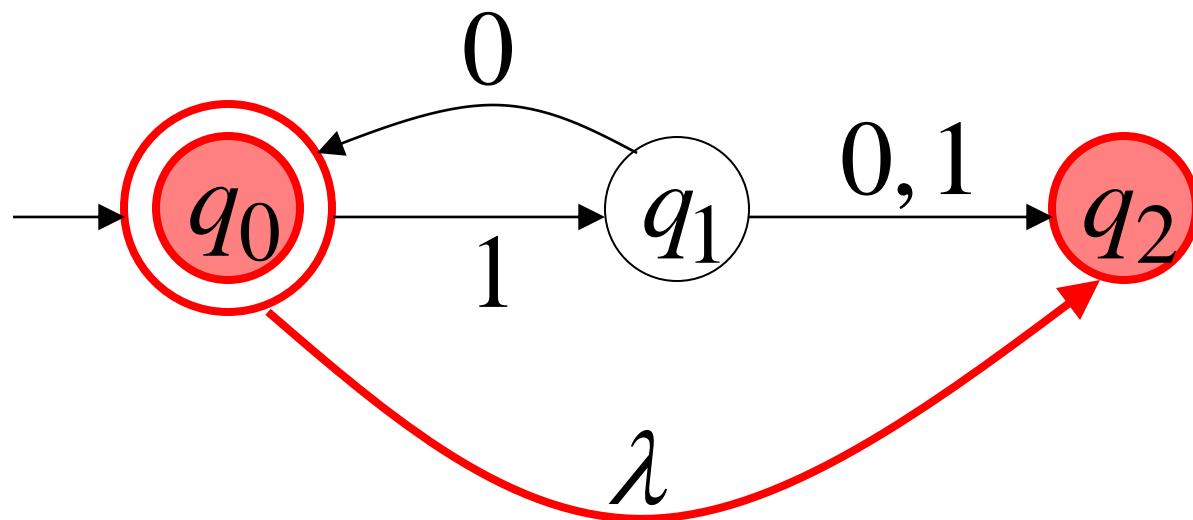
$$\delta(q_0, 1) = \{q_1\}$$



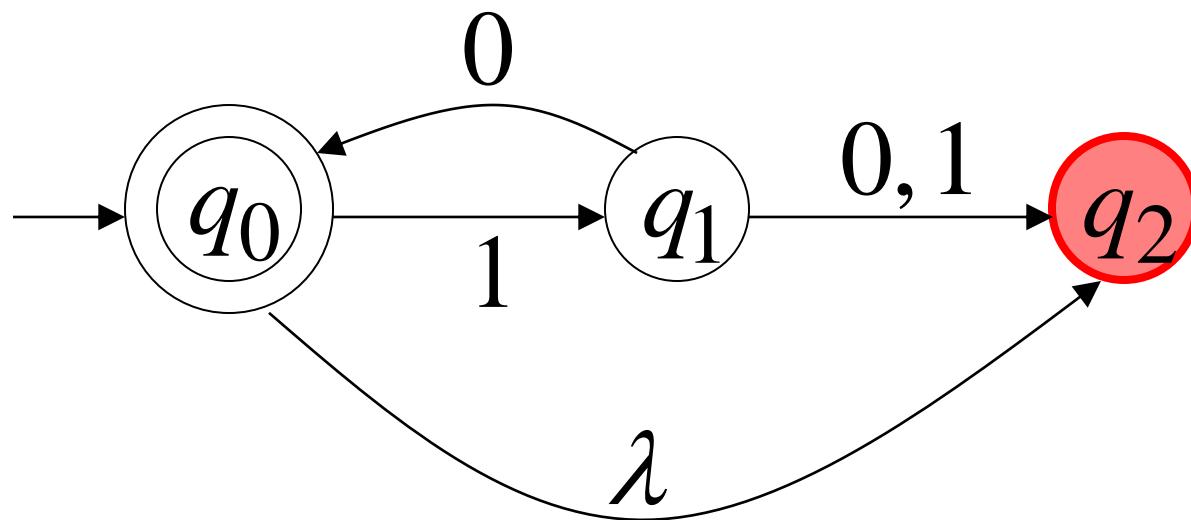
$$\delta(q_1, 0) = \{q_0, q_2\}$$



$$\delta(q_0, \lambda) = \{q_2\}$$



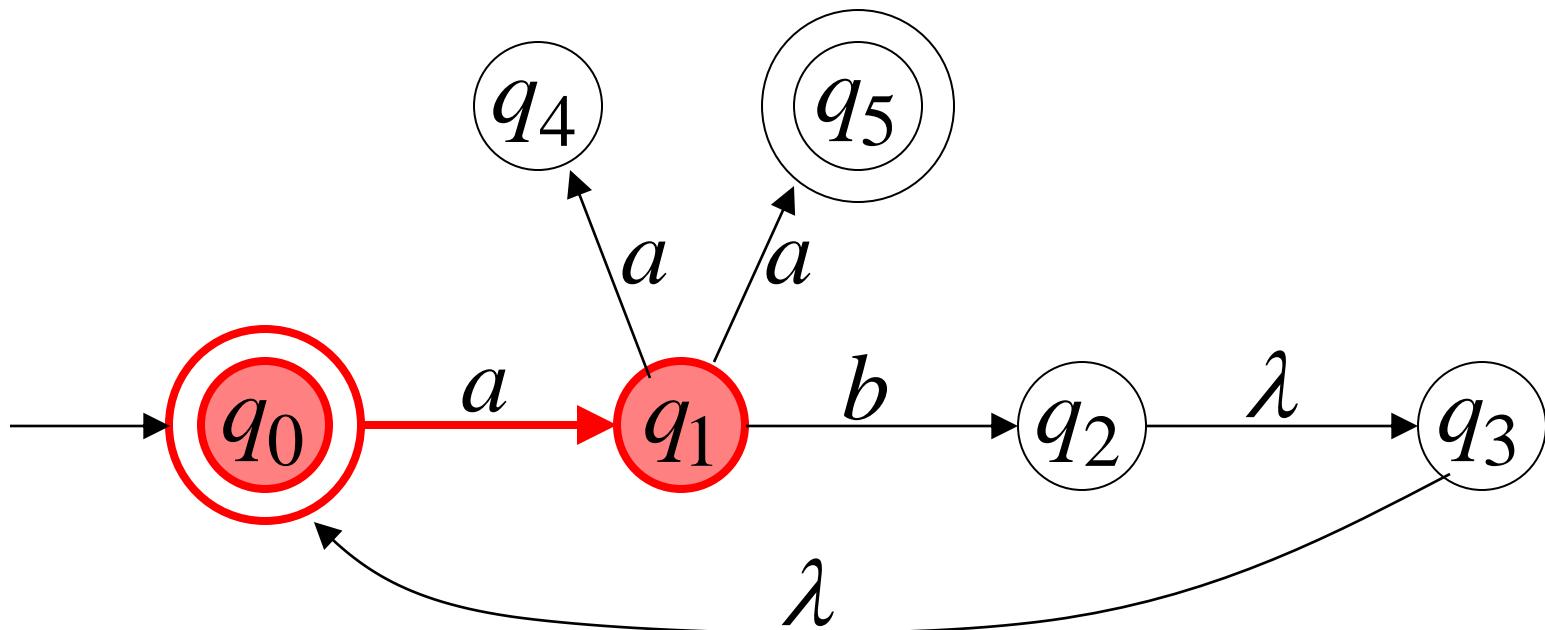
$$\delta(q_2, 1) = \emptyset$$



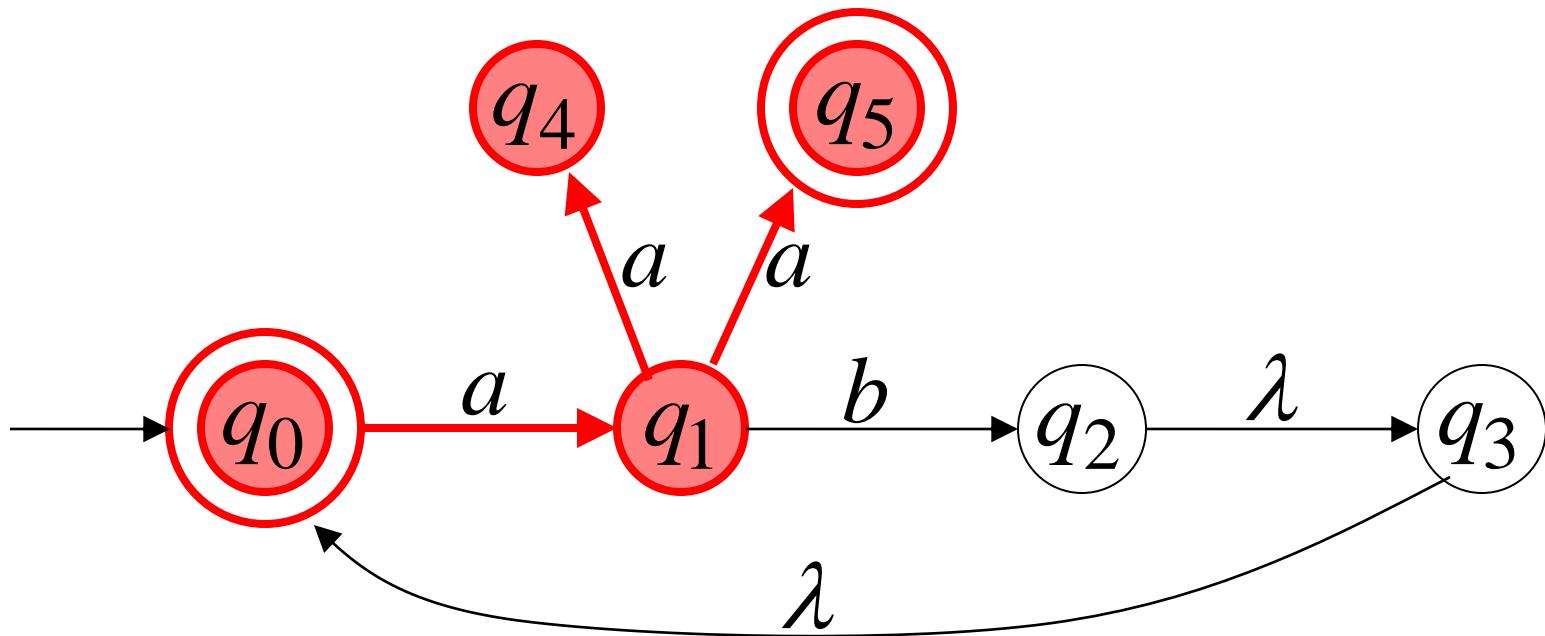
# Extended Transition Function $\delta^*$

Same with  $\delta$  but applied on strings

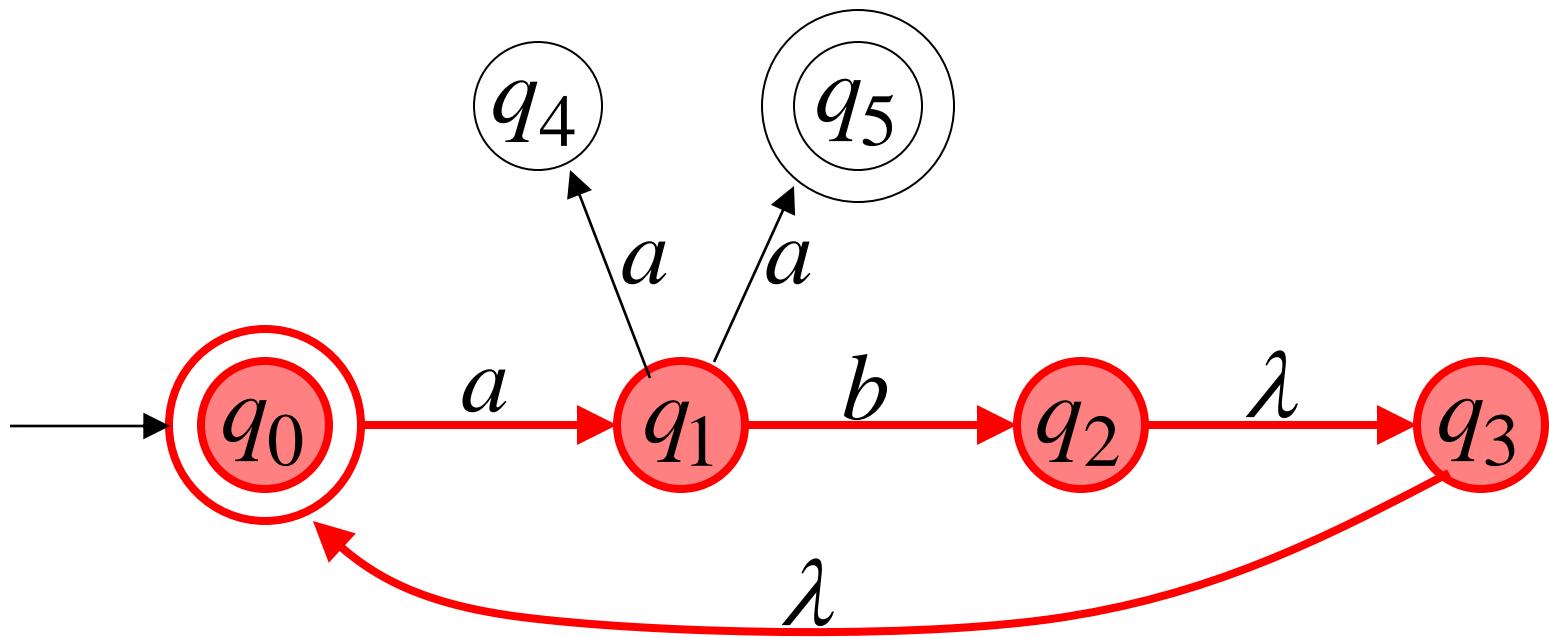
$$\delta^*(q_0, a) = \{q_1\}$$



$$\delta^*(q_0, aa) = \{q_4, q_5\}$$



$$\delta^*(q_0, ab) = \{q_2, q_3, q_0\}$$



Special case:

for any state  $q$

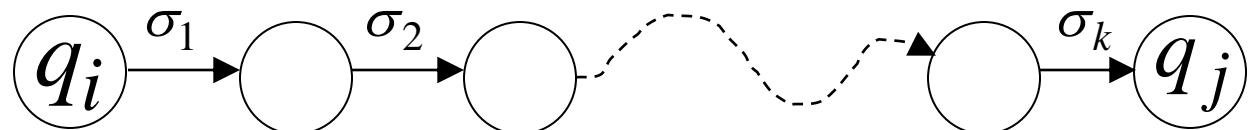
$$q \in \delta^*(q, \lambda)$$

In general

$q_j \in \delta^*(q_i, w)$  : there is a walk from  $q_i$  to  $q_j$   
with label  $w$



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



# The Language of an NFA $M$

The language accepted by  $M$  is:

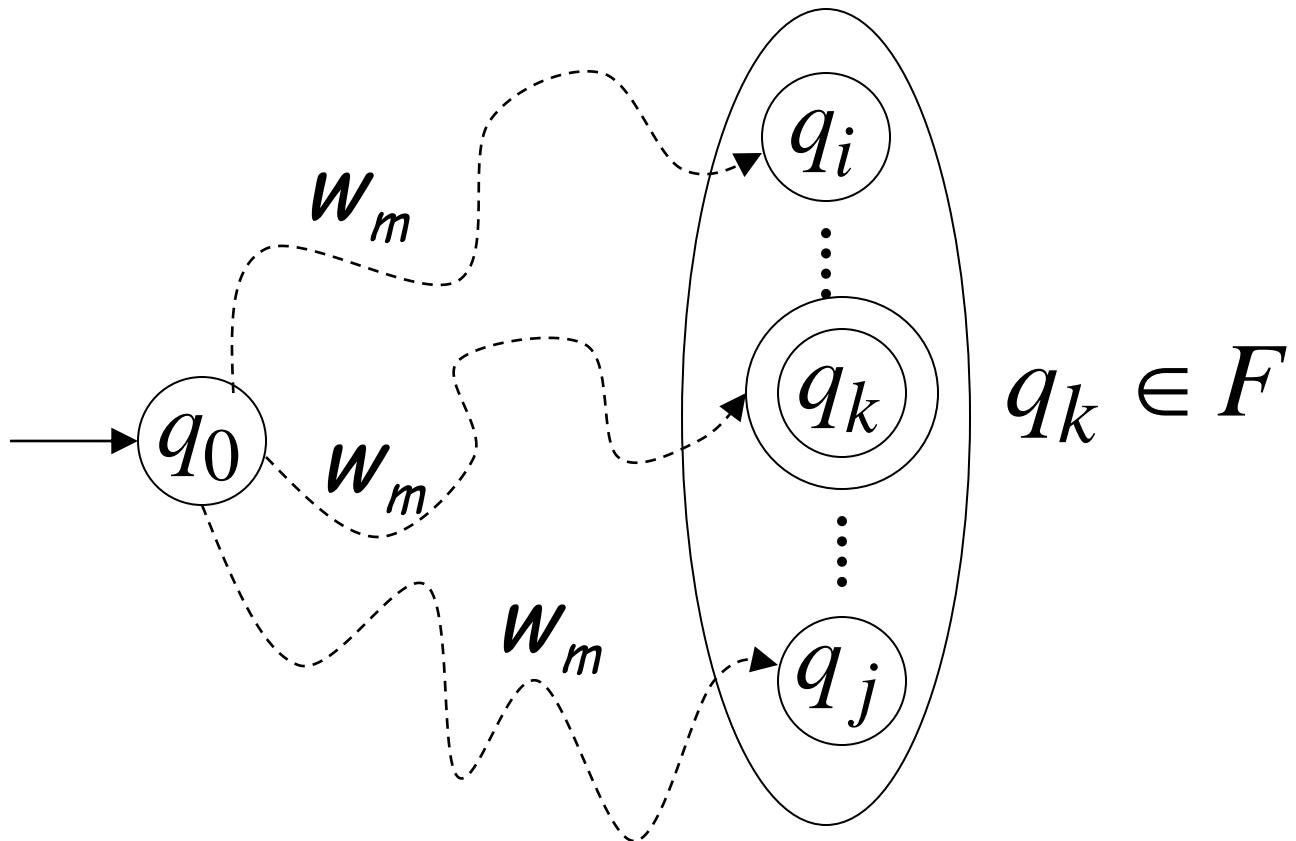
$$L(M) = \{w_1, w_2, \dots, w_n\}$$

where  $\delta^*(q_0, w_m) = \{q_i, \dots, q_k, \dots, q_j\}$

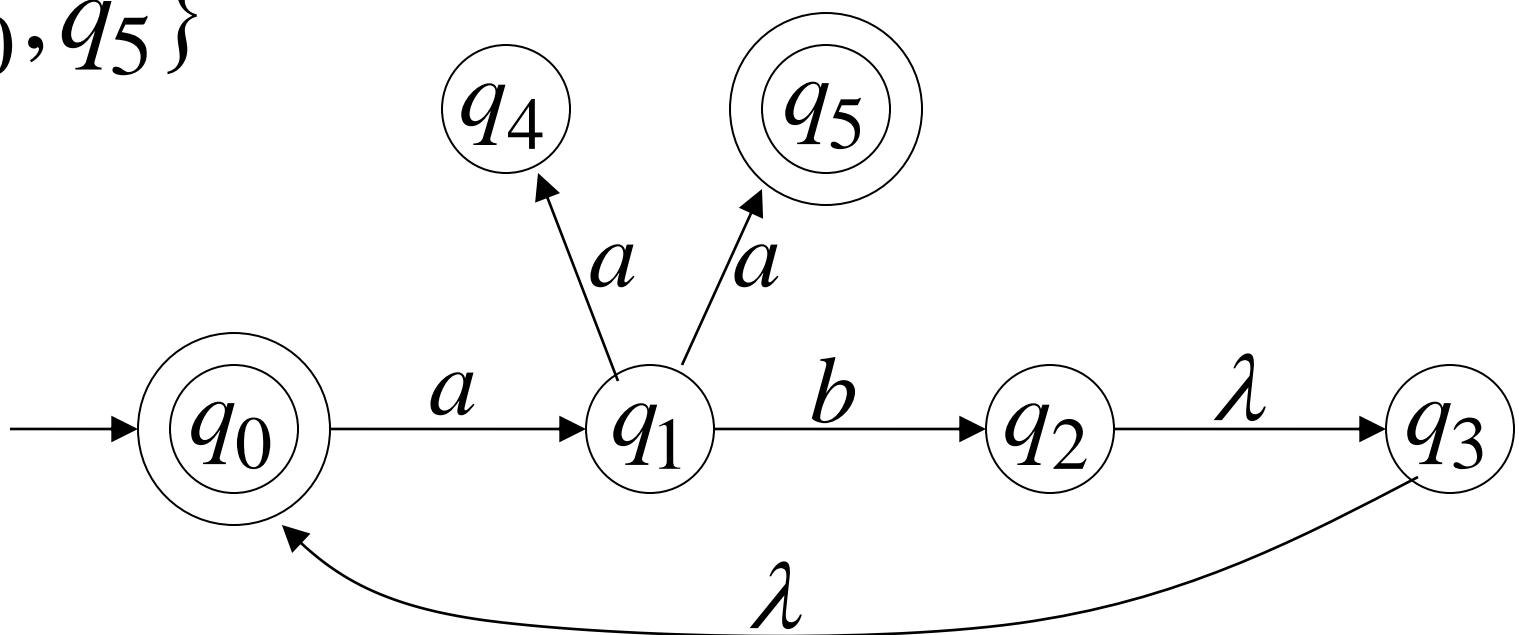
and there is some  $q_k \in F$  (accepting state)

$$w_m \in L(M)$$

$$\delta^*(q_0, w_m)$$



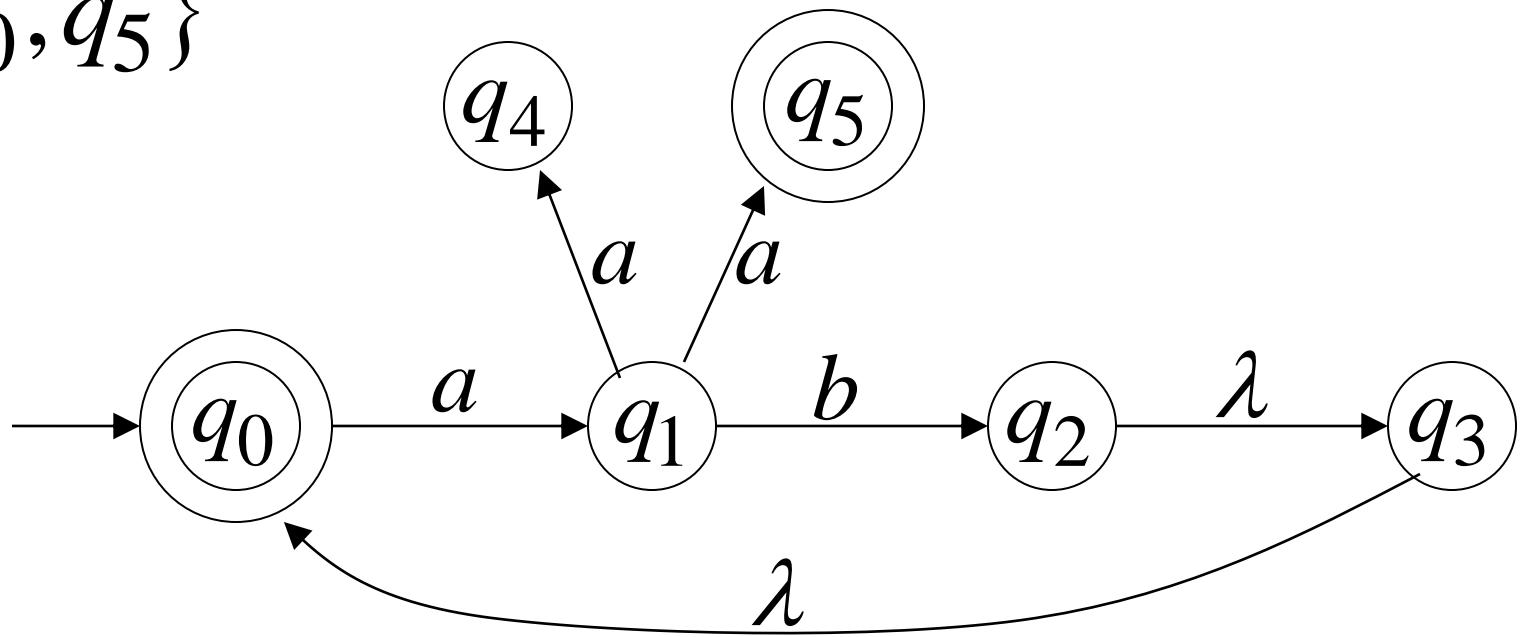
$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, aa) = \{q_4, \underline{q_5}\} \quad \xrightarrow{\hspace{1cm}} \quad aa \in L(M)$$

$\searrow \in F$

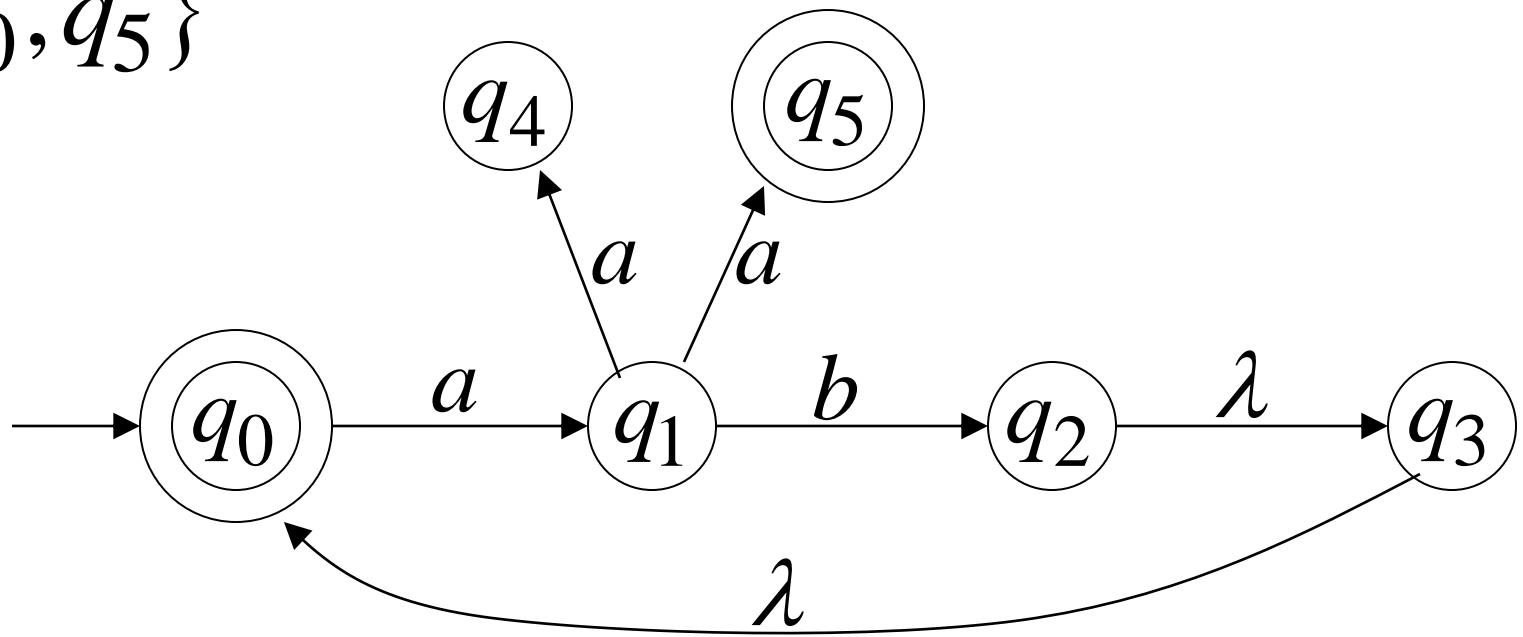
$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, ab) = \{q_2, q_3, \underline{q_0}\} \longrightarrow ab \in L(M)$$

$\searrow \in F$

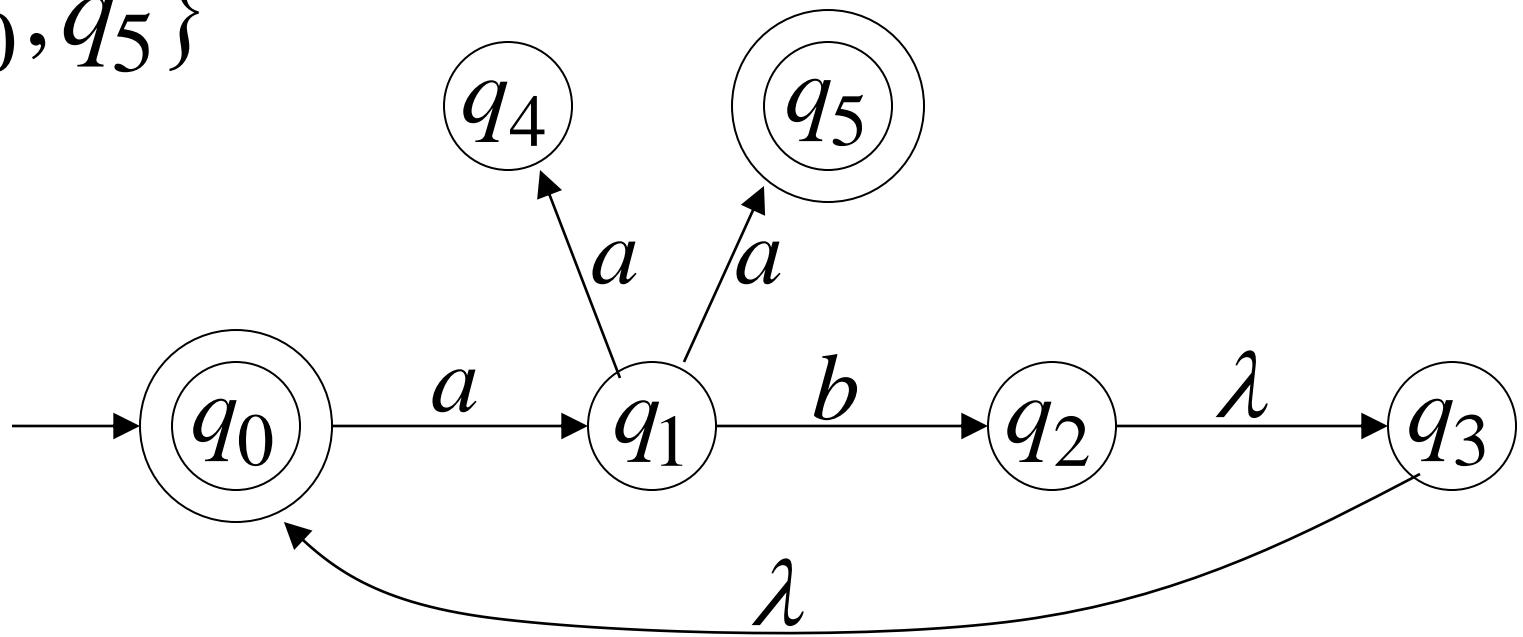
$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, aba) = \{q_4, \underline{q_5}\} \xrightarrow{\quad \text{yellow arrow} \quad} aaba \in L(M)$$

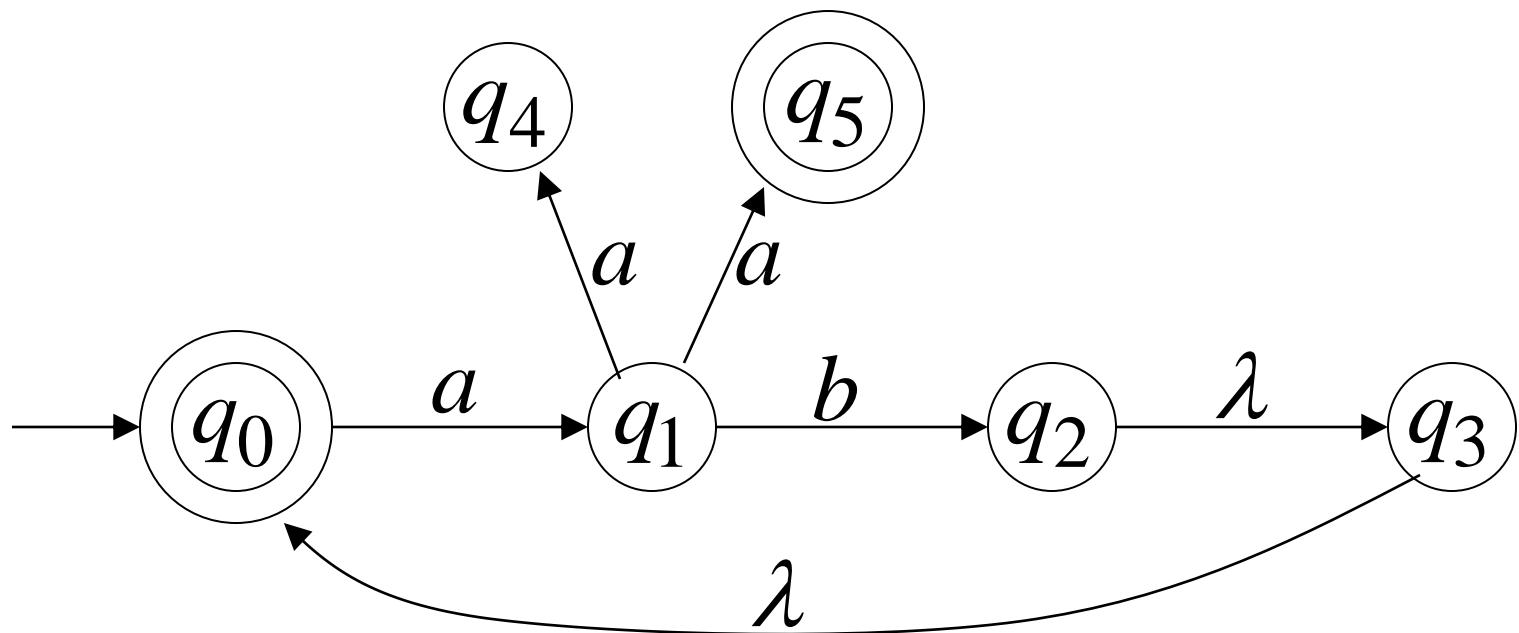
$\xrightarrow{\quad \text{yellow arrow} \quad} \in F$

$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, aba) = \{q_1\} \quad \text{---} \longrightarrow \quad aba \notin L(M)$$

$\not\in F$



$$L(M) = \{ab\}^* \cup \{ab\}^* \{aa\}$$

# Summary

- DFA's, NFA's, and  $\epsilon$ -NFA's all accept exactly the same set of languages: the regular languages.
- The NFA types are easier to design and may have exponentially fewer states than a DFA.

NFAs accept the Regular  
Languages

# Equivalence of Machines

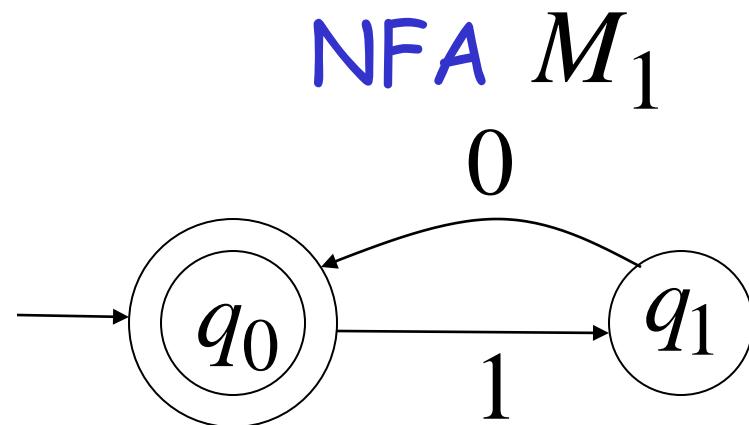
Definition:

Machine  $M_1$  is equivalent to machine  $M_2$

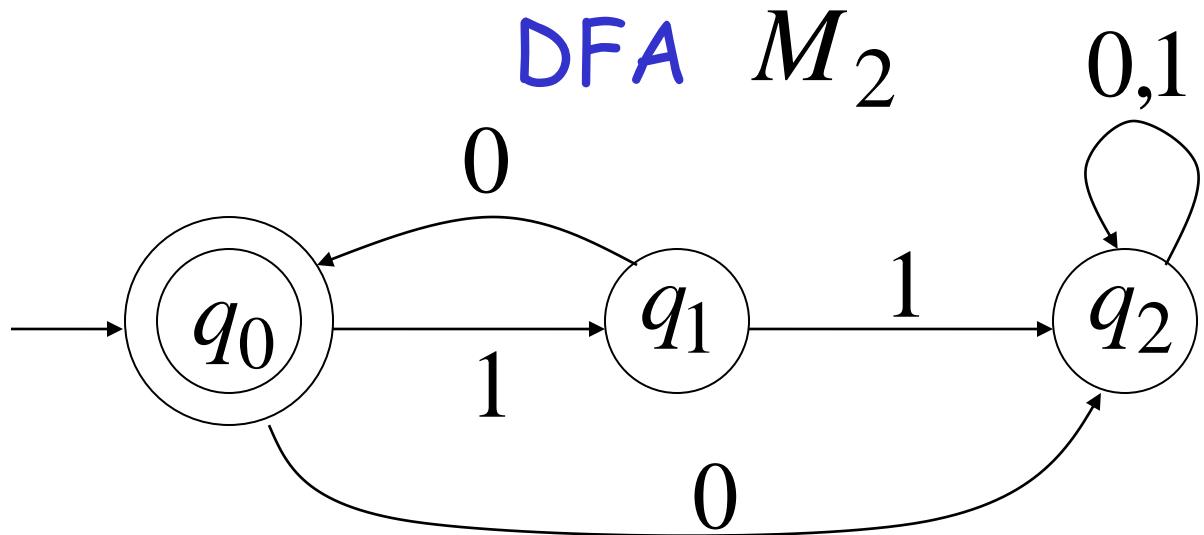
if  $L(M_1) = L(M_2)$

# Example of equivalent machines

$$L(M_1) = \{10\}^*$$



$$L(M_2) = \{10\}^*$$



# Theorem:

$$\left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by NFAs} \end{array} \right\} = \left\{ \begin{array}{l} \text{Regular} \\ \text{Languages} \end{array} \right\}$$

Languages  
accepted  
by DFAs

NFAs and DFAs have the same computation power,  
accept the same set of languages

**Proof:** we only need to show

$$\left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by NFAs} \end{array} \right\} \supseteq \left\{ \begin{array}{l} \text{Regular} \\ \text{Languages} \end{array} \right\}$$

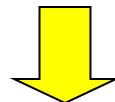
AND

$$\left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by NFAs} \end{array} \right\} \subseteq \left\{ \begin{array}{l} \text{Regular} \\ \text{Languages} \end{array} \right\}$$

## Proof-Step 1

$$\left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by NFAs} \end{array} \right\} \equiv \left\{ \begin{array}{l} \text{Regular} \\ \text{Languages} \end{array} \right\}$$

Every DFA is trivially an NFA

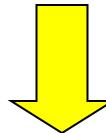


Any language  $L$  accepted by a DFA  
is also accepted by an NFA

## Proof-Step 2

$$\left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by NFAs} \end{array} \right\} \subseteq \left\{ \begin{array}{l} \text{Regular} \\ \text{Languages} \end{array} \right\}$$

Any NFA can be converted to an equivalent DFA

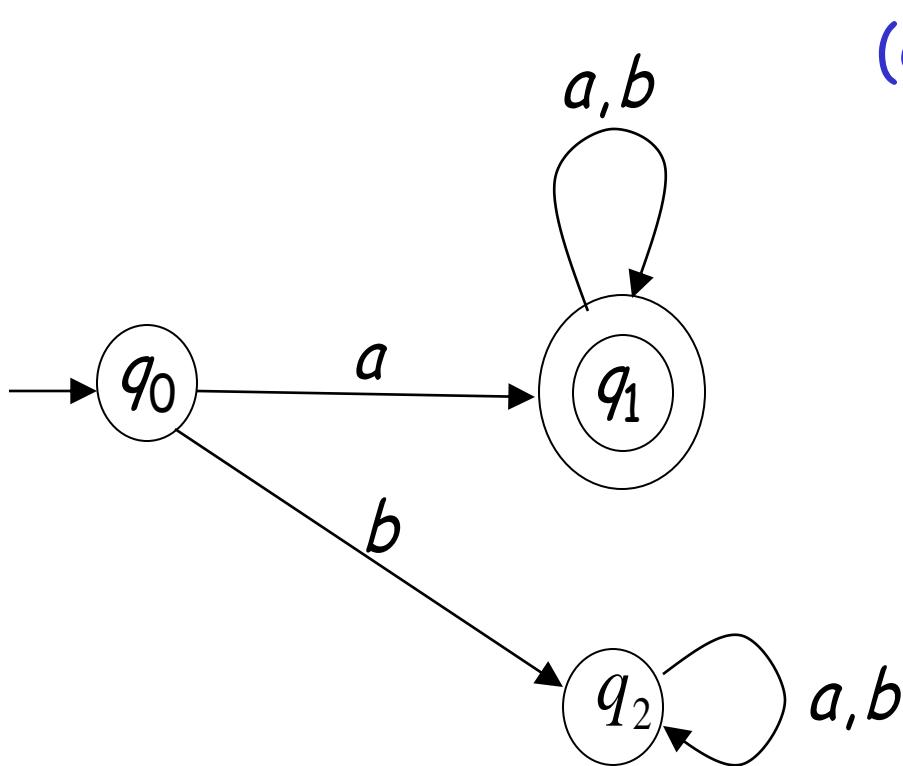


Any language  $L$  accepted by an NFA is also accepted by a DFA

# DFA Minimization

# Minimization of FA

- Input - DFA
- Output - Minimized DFA
- Step 1 - Draw a table for all pairs of states  $(Q_i, Q_j)$  ( $i \neq j$ )  
[All are unmarked initially]

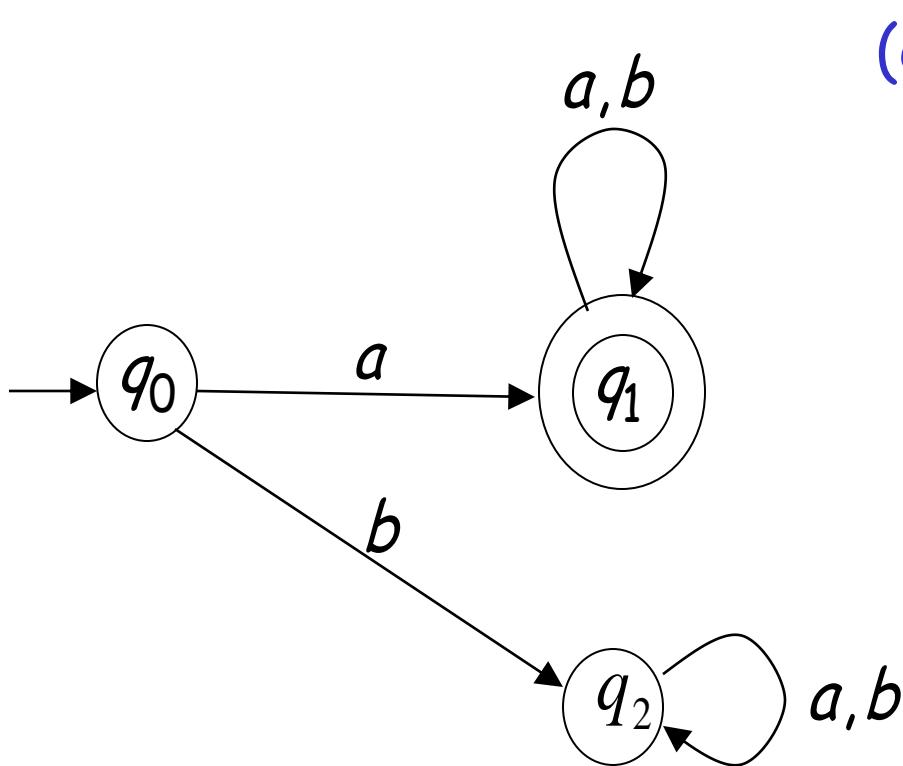


(q0,q1)    (q0,q2)    (q1,q2)

q0			
q1			
q2			
	q0	q1	q2

# Minimization of FA

- Input - DFA
- Output - Minimized DFA
- Step 1 - Draw a table for all pairs of states  $(Q_i, Q_j)$  ( $i \neq j$ )  
[All are unmarked initially]



(q0,q1)    (q0,q2)    (q1,q2)

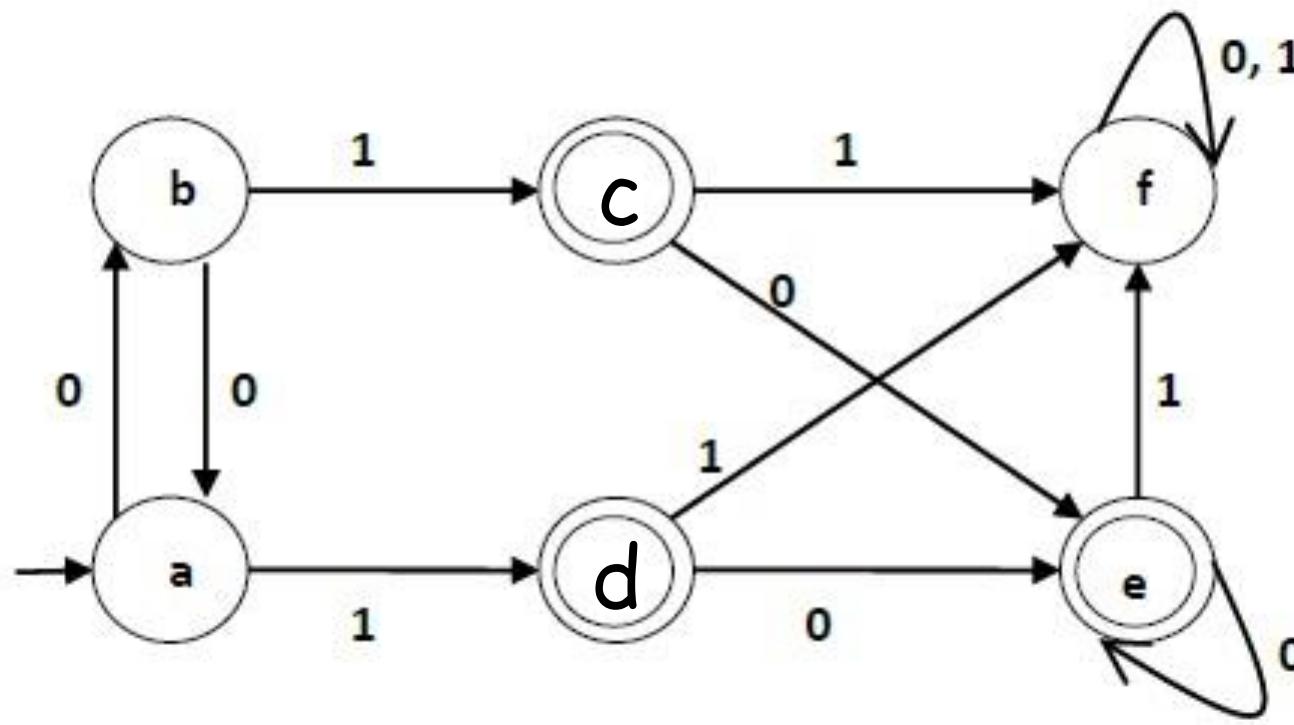
q1	
q2	
	q0    q1

# Minimization of FA

- **Input - DFA**
- **Output - Minimized DFA**
- **Step 1** - Draw a table for all pairs of states  $(Q_i, Q_j)$  ( $i \neq j$ )  
[All are unmarked initially]
- **Step 2** - Consider every state pair  $(Q_i, Q_j)$  in the DFA where  $Q_i \in F$  and  $Q_j \notin F$  or vice versa and mark them.  
[Here  $F$  is the set of final states]
- **Step 3** - Repeat this step until we cannot mark anymore states -  

If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.
- **Step 4** - Combine all the unmarked pair  $(Q_i, Q_j)$  and make them a single state in the reduced DFA.

# Example

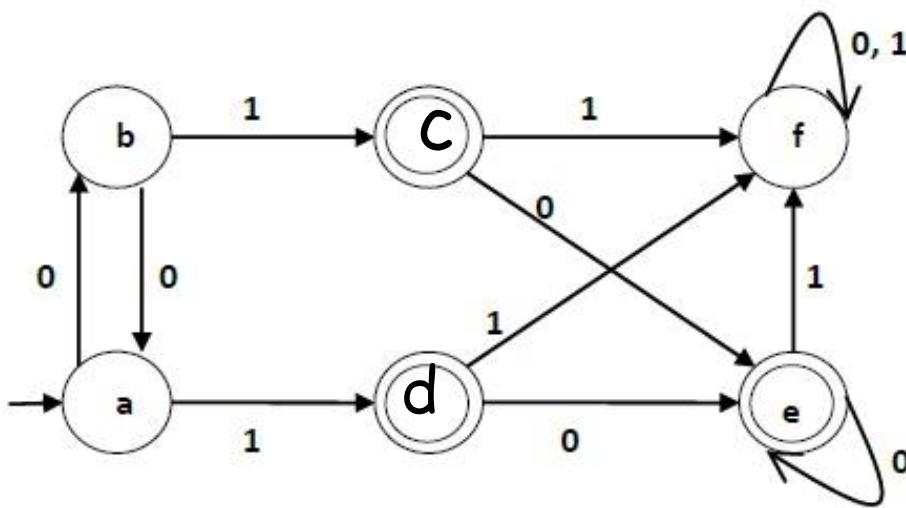


- Step 1 - Draw a table for all pairs of states ( $Q_i, Q_j$ ) [All are unmarked initially].

a						
b						
c						
d						
e						
f						
	a	b	c	d	e	F

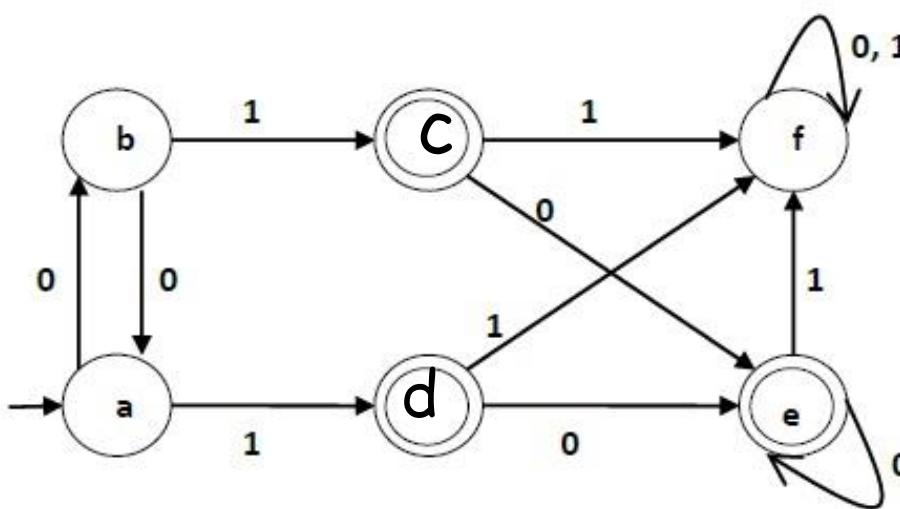
- Step 1 – Draw a table for all pairs of states ( $Q_i, Q_j$ ) [All are unmarked initially].

a						
b						
c						
d						
e						
f						
	a	b	c	d	e	F



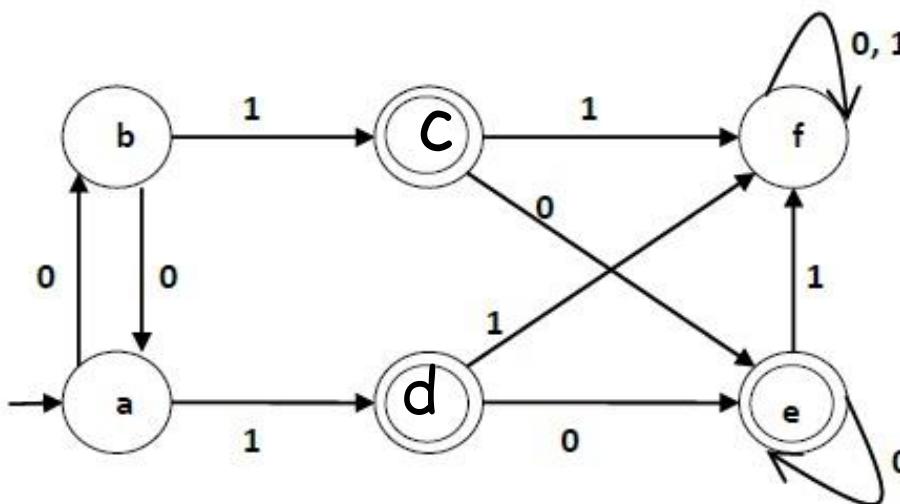
- Step 2 - Consider every state pair  $(Q_i, Q_j)$  in the DFA where  $Q_i \in F$  and  $Q_j \notin F$  or vice versa and mark them. [Here  $F$  is the set of final states] If both the states from pair are final then don't mark them.

a						
b						
c						
d						
e						
f						
	a	b	c	d	e	F



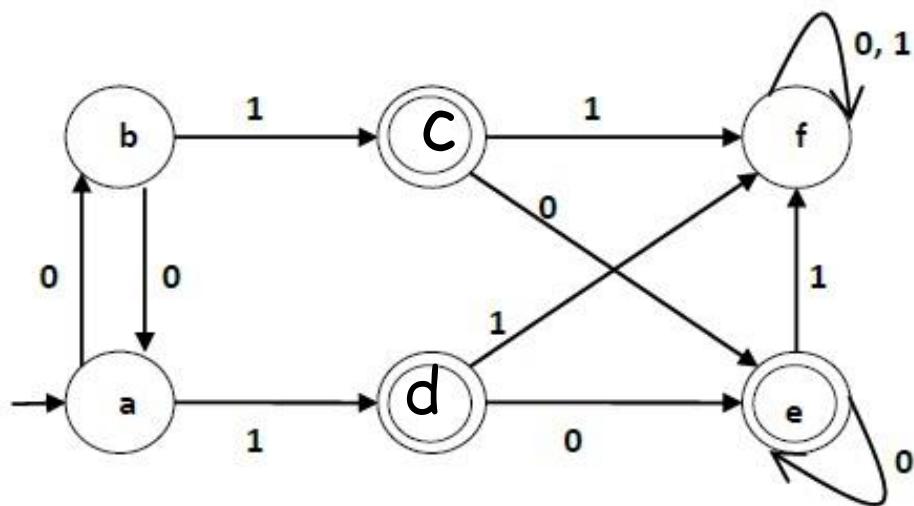
- Step 2 - Consider every state pair  $(Q_i, Q_j)$  in the DFA where  $Q_i \in F$  and  $Q_j \notin F$  or vice versa and mark them. [Here  $F$  is the set of final states]. If both the states from pair are final then don't mark them.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f			1	1	1	
	a	B	c	D	e	f



- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f			1	1	1	
	a	b	c	d	e	f



Unmarked pair is (a,f)

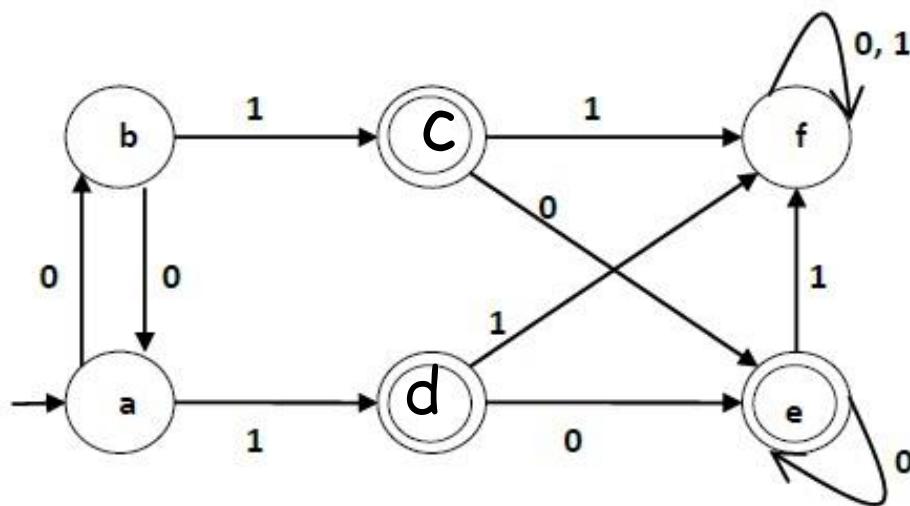
$$\begin{aligned}\delta(a, 0) &= b & (b, f) \\ \delta(f, 0) &= f\end{aligned}$$

Resultant pair are marked

$$\begin{aligned}\delta(a, 1) &= d & (d, f) \\ \delta(f, 1) &= f\end{aligned}$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2		1	1	1	
	a	b	c	d	e	f



Unmarked pair is (a,f)

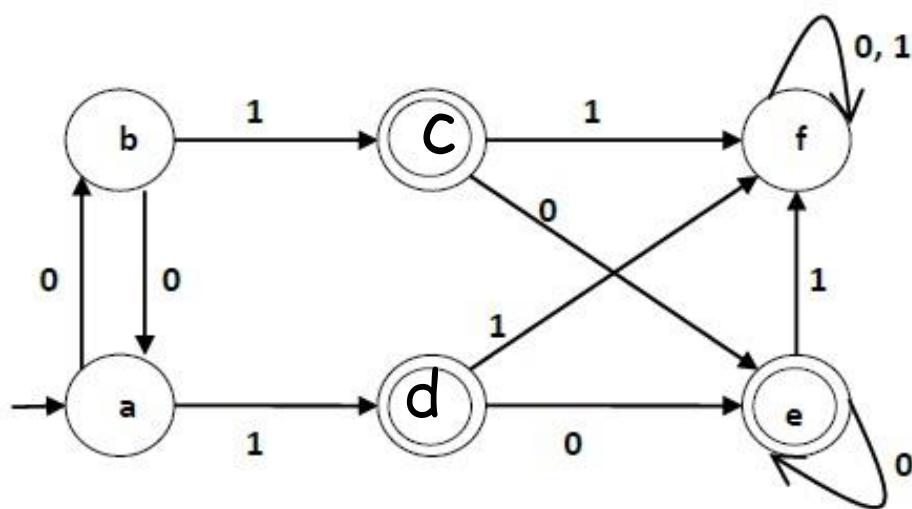
$$\begin{aligned}\delta(a, 0) &= b & (b, f) \\ \delta(f, 0) &= f\end{aligned}$$

Resultant pair are marked

$$\begin{aligned}\delta(a, 1) &= d & (d, f) \\ \delta(f, 1) &= f\end{aligned}$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2		1	1	1	
	a	b	c	d	e	f



Unmarked pair is (a,b)

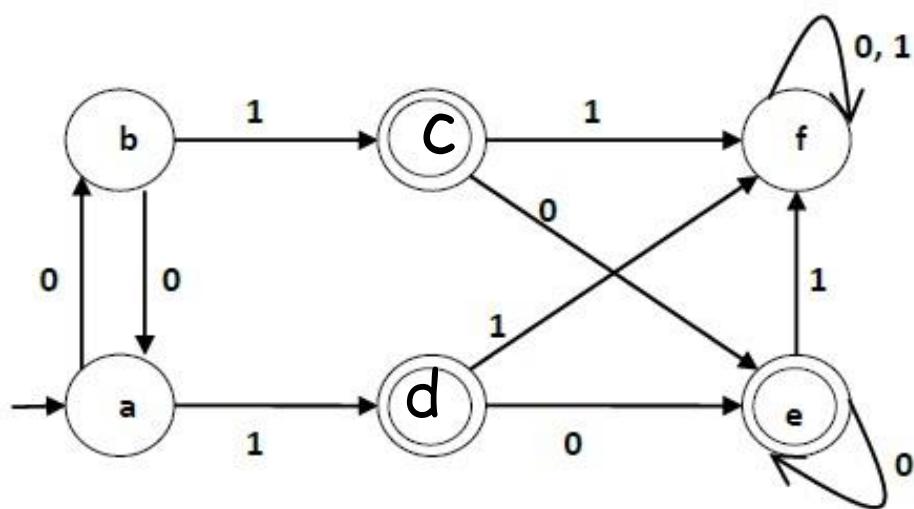
$$\begin{aligned}\delta(a, 0) &= b \\ \delta(b, 0) &= a\end{aligned}$$

Resultant pair are unmarked

$$\begin{aligned}\delta(a, 1) &= d \\ \delta(b, 1) &= c\end{aligned}$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2		1	1	1	
	a	b	c	d	e	f



Unmarked pair is  $(b, f)$

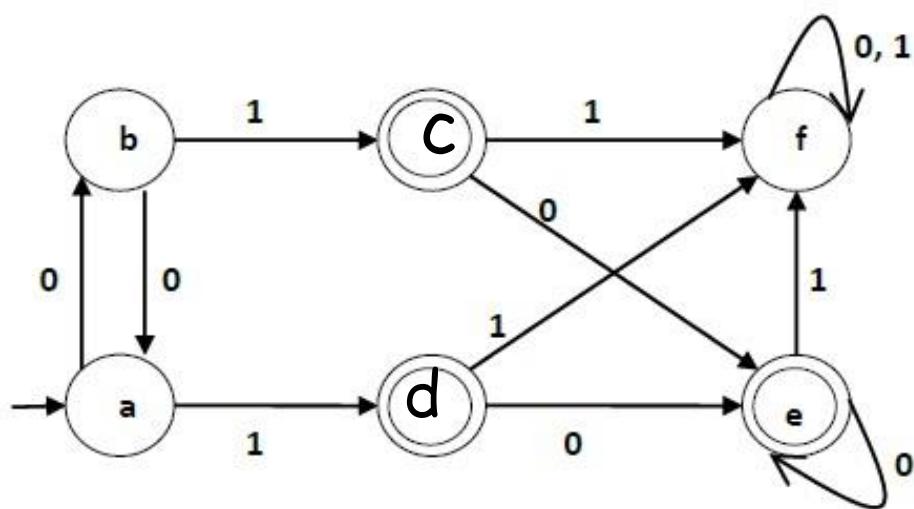
$$\begin{aligned}\delta(b, 0) &= a & (a, f) \\ \delta(f, 0) &= f\end{aligned}$$

Resultant pair are marked

$$\begin{aligned}\delta(b, 1) &= c & (c, f) \\ \delta(f, 1) &= f\end{aligned}$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2	2	1	1	1	
	a	b	c	d	e	f



Unmarked pair is  $(b, f)$

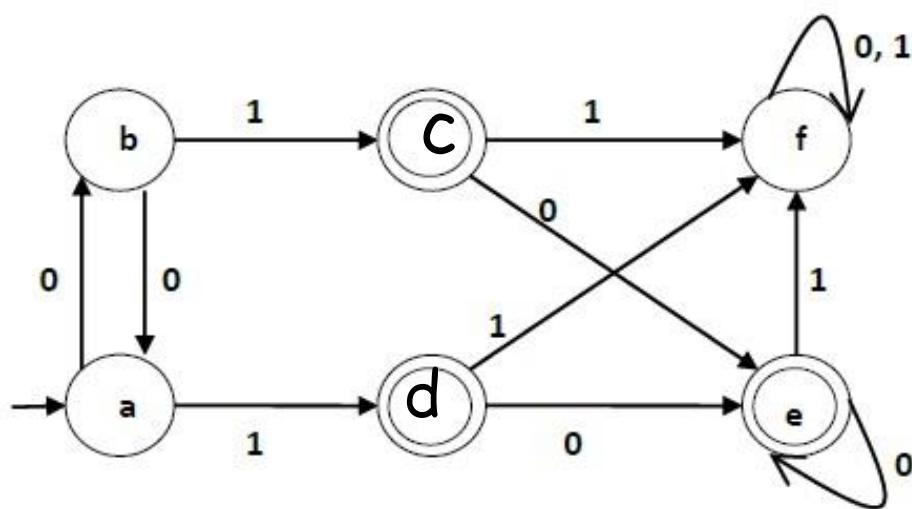
$$\begin{aligned}\delta(b, 0) &= a & (a, f) \\ \delta(f, 0) &= f\end{aligned}$$

Resultant pair are marked

$$\begin{aligned}\delta(b, 1) &= c & (c, f) \\ \delta(f, 1) &= f\end{aligned}$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2	2	1	1	1	
	a	b	c	d	e	f



Unmarked pair is  $(c, e)$

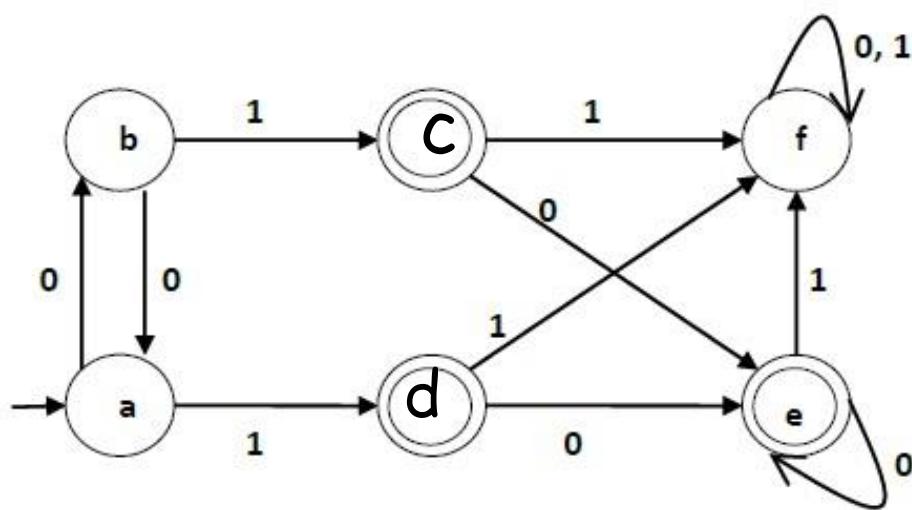
$$\begin{aligned}\delta(c, 0) &= e \\ \delta(e, 0) &= e\end{aligned}$$

Resultant pair are unmarked

$$\begin{aligned}\delta(c, 1) &= f \\ \delta(e, 1) &= f\end{aligned}$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2	2	1	1	1	
	a	b	c	d	e	f



Unmarked pair is  $(c,d)$

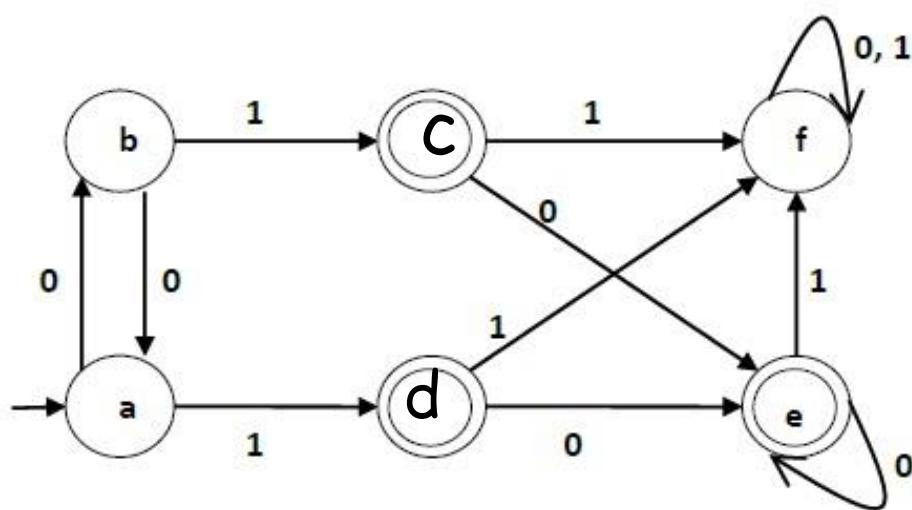
$$\begin{aligned}\delta(c, 0) &= e && (e, e) \\ \delta(d, 0) &= e\end{aligned}$$

Resultant pair are unmarked

$$\begin{aligned}\delta(c, 1) &= f && (f, f) \\ \delta(d, 1) &= f\end{aligned}$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2	2	1	1	1	
	a	b	c	d	e	f



Unmarked pair is  $(d, e)$

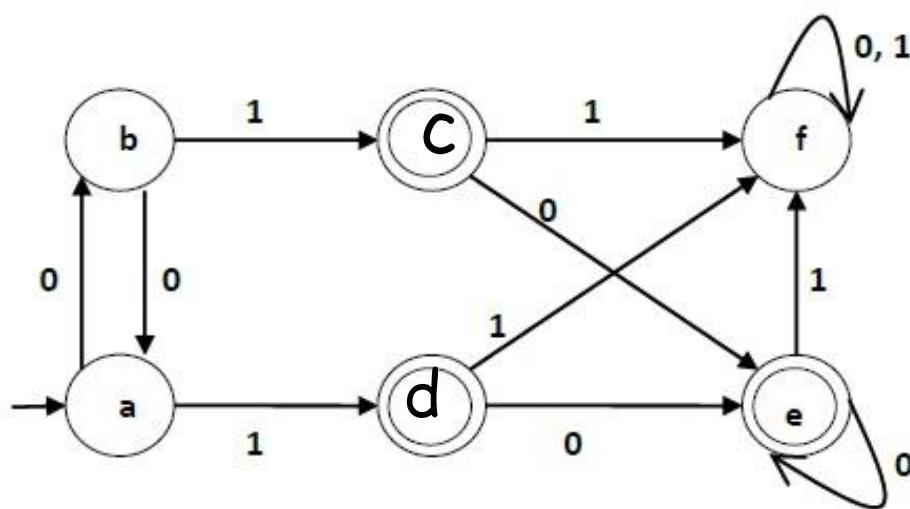
$$\begin{aligned}\delta(d, 0) &= e \\ \delta(e, 0) &= e\end{aligned}$$

Resultant pair are unmarked

$$\begin{aligned}\delta(d, 1) &= f \\ \delta(e, 1) &= f\end{aligned}$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2	2	1	1	1	
	a	b	c	d	e	f



Unmarked pair is (a,b)

$$\delta(a, 0) = b \quad (b, a)$$

$$\delta(b, 0) = a$$

Resultant pair are unmarked

$$\delta(a, 1) = d \quad (d, c)$$

$$\delta(b, 1) = c$$

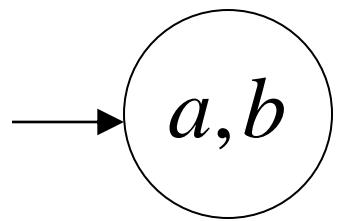
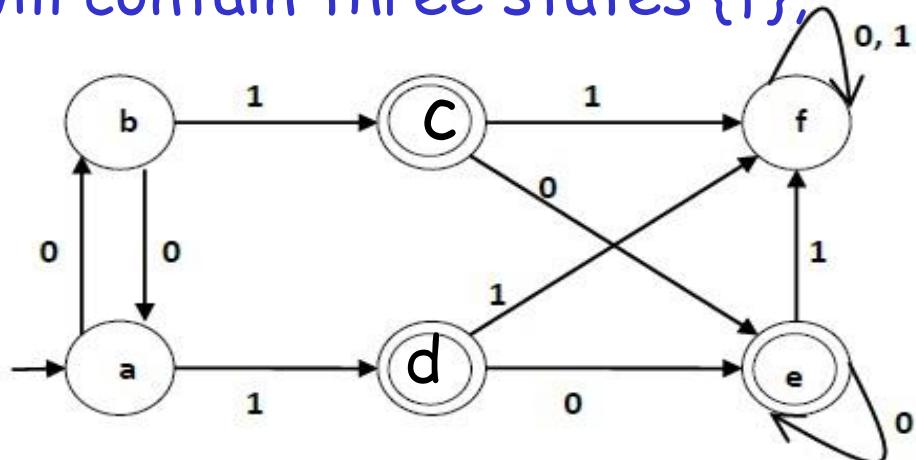
- Step 4 - Combine all the unmarked pair ( $Q_i, Q_j$ ) and make them a single state in the reduced DFA.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2	2	1	1	1	
	a	b	c	d	e	f

- After step 3, we have got state combinations  $\{a, b\} \{c, d\} \{c, e\} \{d, e\}$  that are unmarked.
- We can recombine  $\{c, d\} \{c, e\} \{d, e\}$  into  $\{c, d, e\}$
- Hence we got two combined states as -  $\{a, b\}$  and  $\{c, d, e\}$

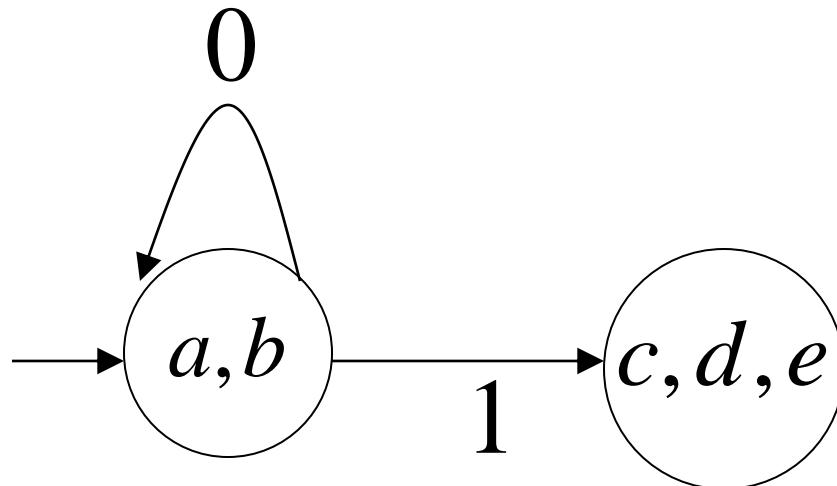
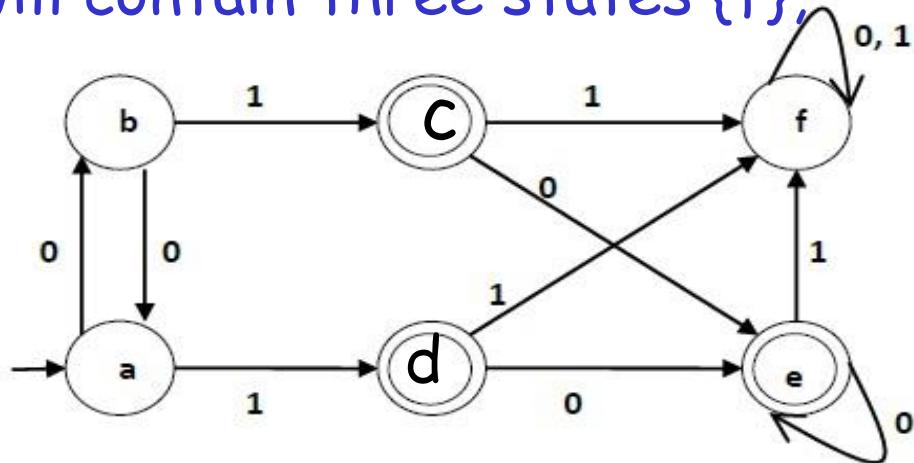
- So the final minimized DFA will contain three states  $\{f\}$ ,  $\{a, b\}$  and  $\{c, d, e\}$

$\{a, b\}$



- So the final minimized DFA will contain three states  $\{f\}$ ,  $\{a, b\}$  and  $\{c, d, e\}$
- $\{a, b\}$

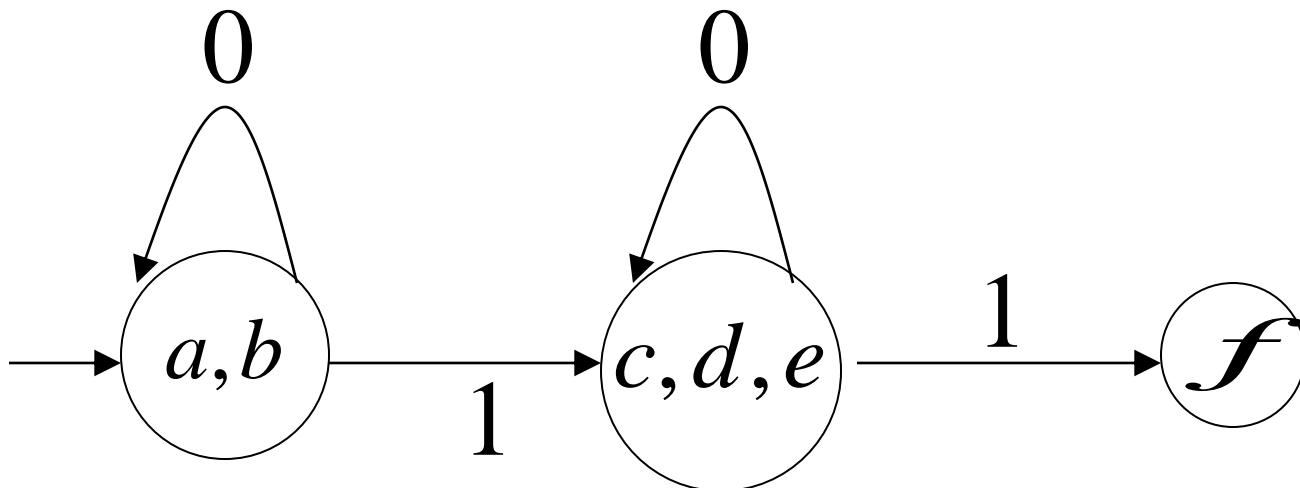
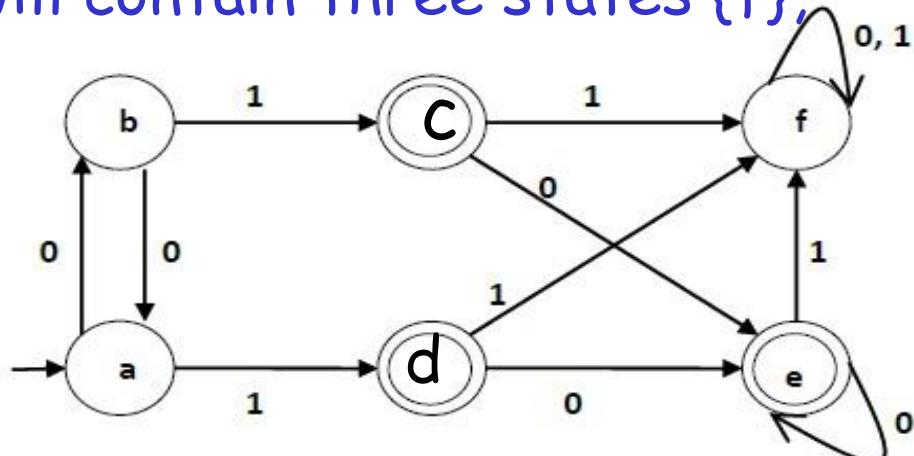
$$\begin{array}{ll} \delta(a, 0) = b & \delta(a, 1) = d \\ \delta(b, 0) = a & \delta(b, 1) = c \end{array}$$



- So the final minimized DFA will contain three states  $\{f\}$ ,  $\{a, b\}$  and  $\{c, d, e\}$

$\{c, d, e\}$

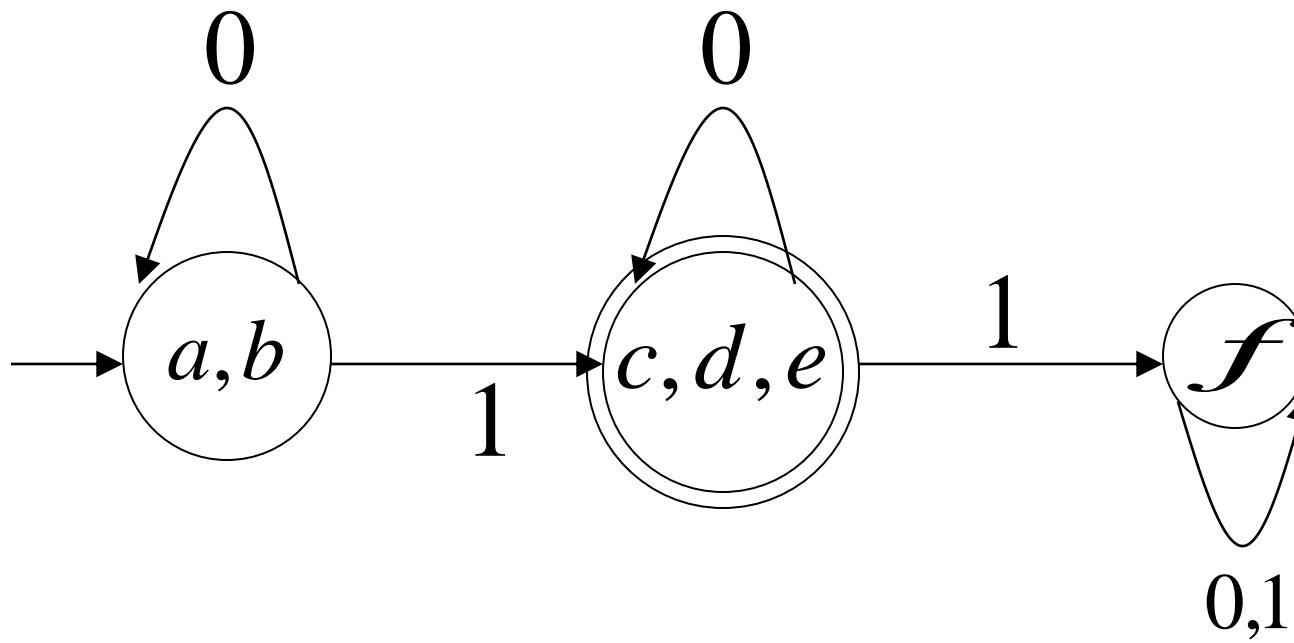
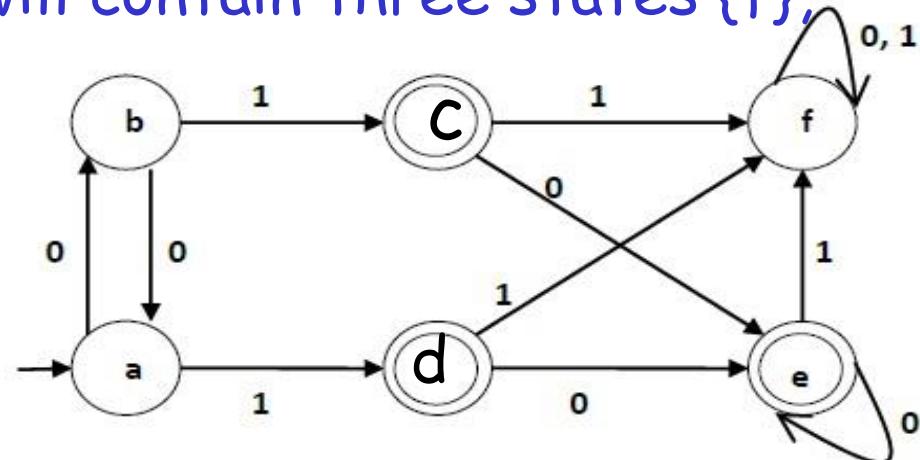
$$\begin{array}{ll} \delta(c, 0) = e & \delta(c, 1) = f \\ \delta(d, 0) = e & \delta(d, 1) = f \\ \delta(e, 0) = e & \delta(e, 1) = f \end{array}$$



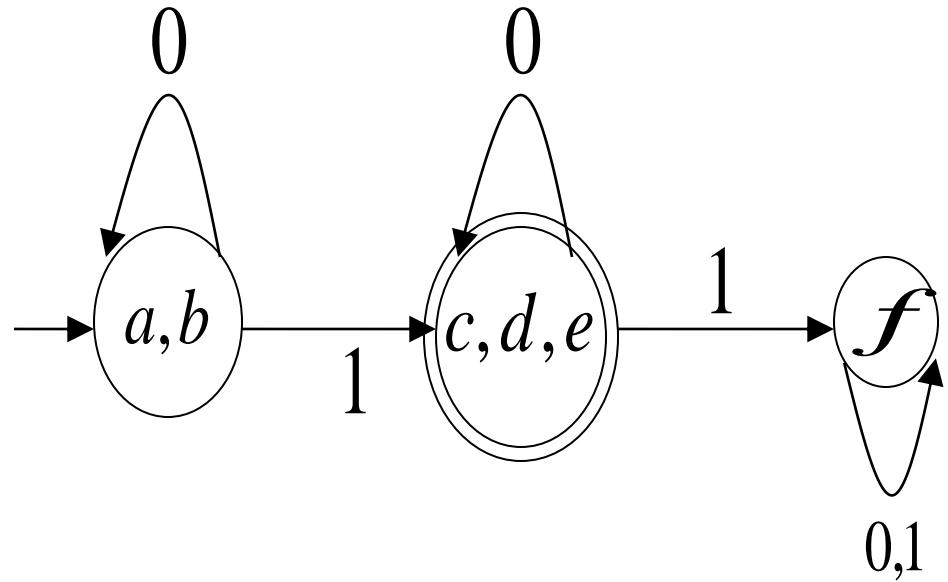
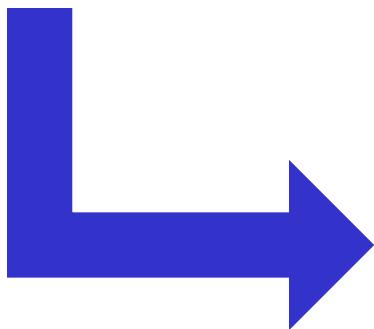
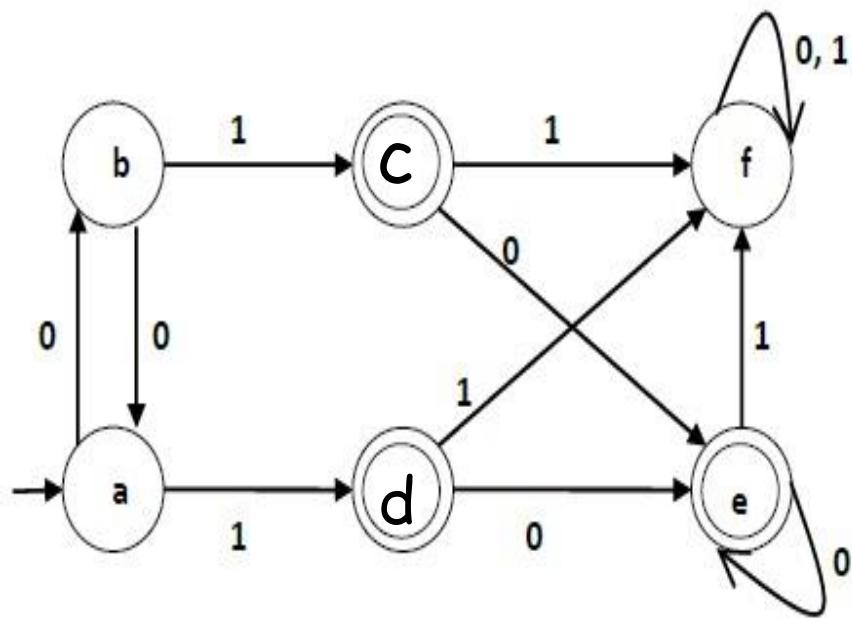
- So the final minimized DFA will contain three states  $\{f\}$ ,  $\{a, b\}$  and  $\{c, d, e\}$

$\{f\}$

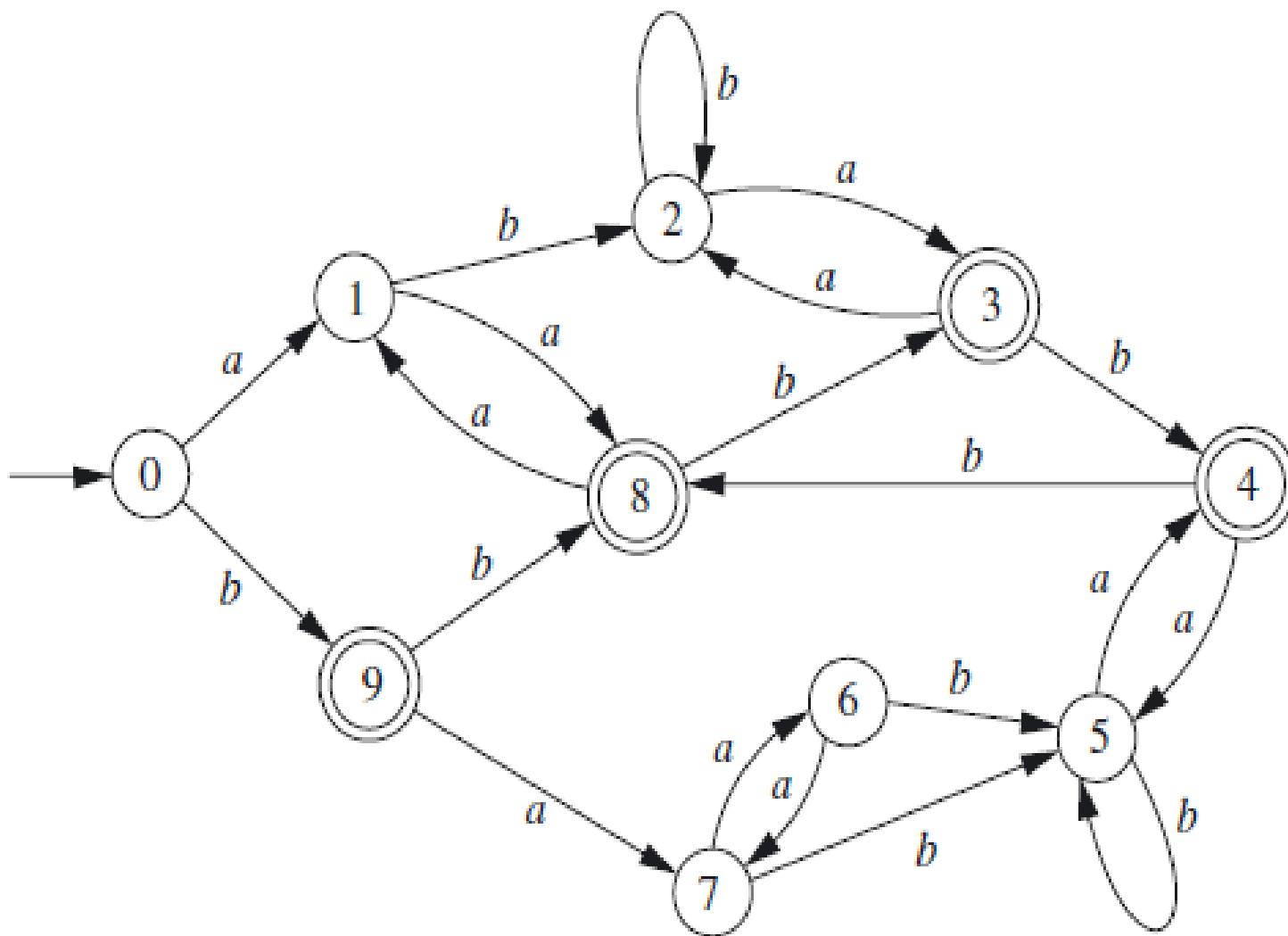
$$\delta(f, 0) = f \quad \delta(f, 1) = f$$



# Example

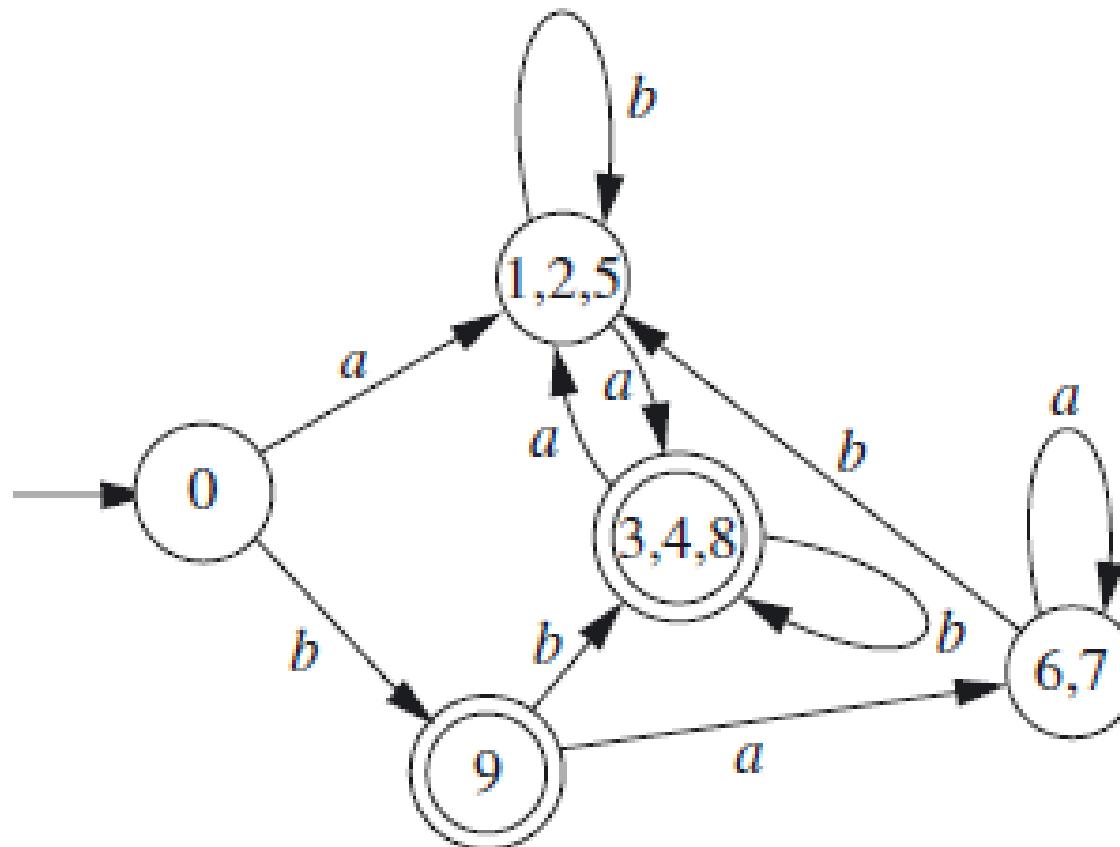


# Minimize following DFA



1	2							
2	2							
3	1	1	1					
4	1	1	1					
5	2			1	1			
6	2	2	2	1	1	2		
7	2	2	2	1	1	2		
8	1	1	1			1	1	1
9	1	1	1	2	3	1	1	2
	0	1	2	3	4	5	6	7

# Final DFA



# Absent No 28/6/19

- 2,7,16,18,21,25,26,27,29,34,37,50,53,57,59,65,70,

# Recall

- NFA
- NFA-null

# The Language of an NFA $M$

The language accepted by  $M$  is:

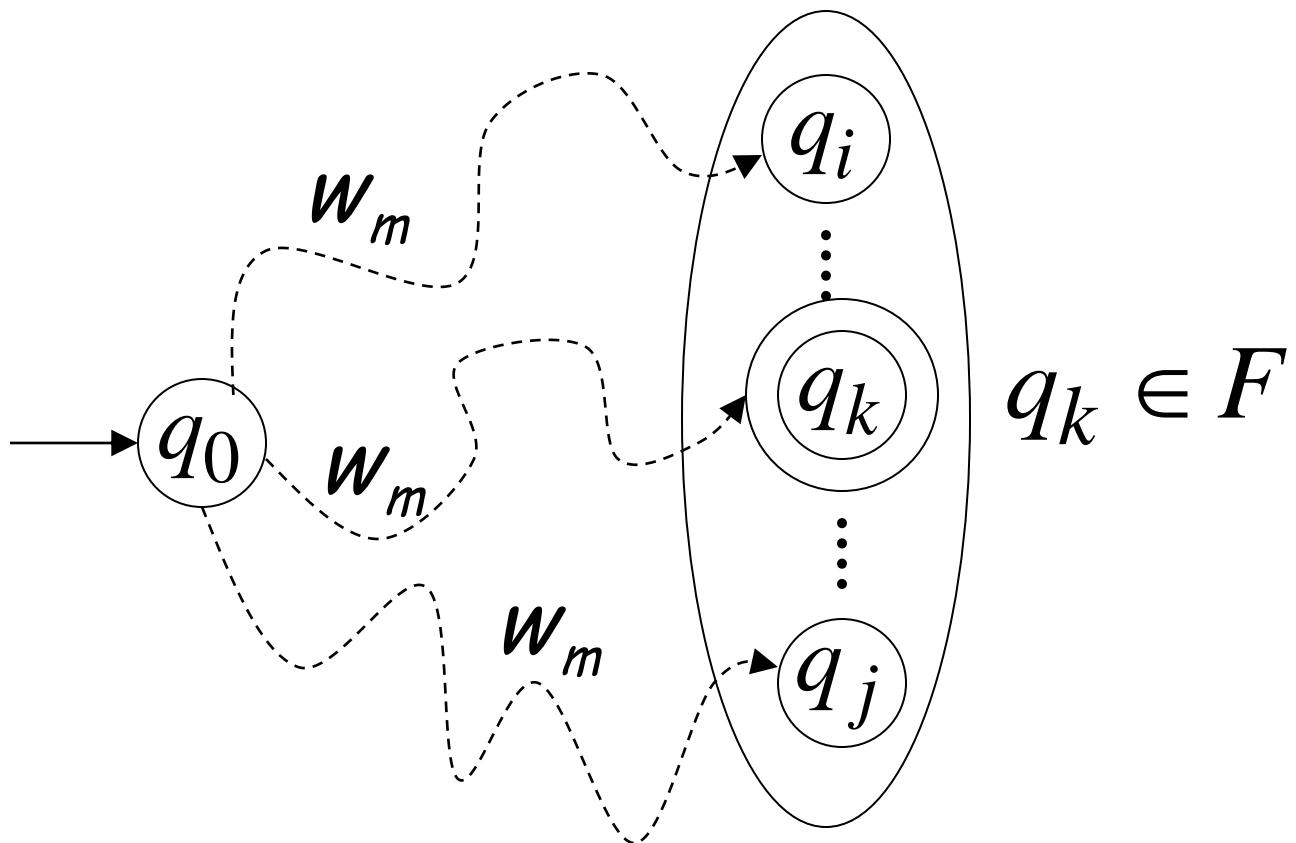
$$L(M) = \{w_1, w_2, \dots, w_n\}$$

where  $\delta^*(q_0, w_m) = \{q_i, \dots, q_k, \dots, q_j\}$

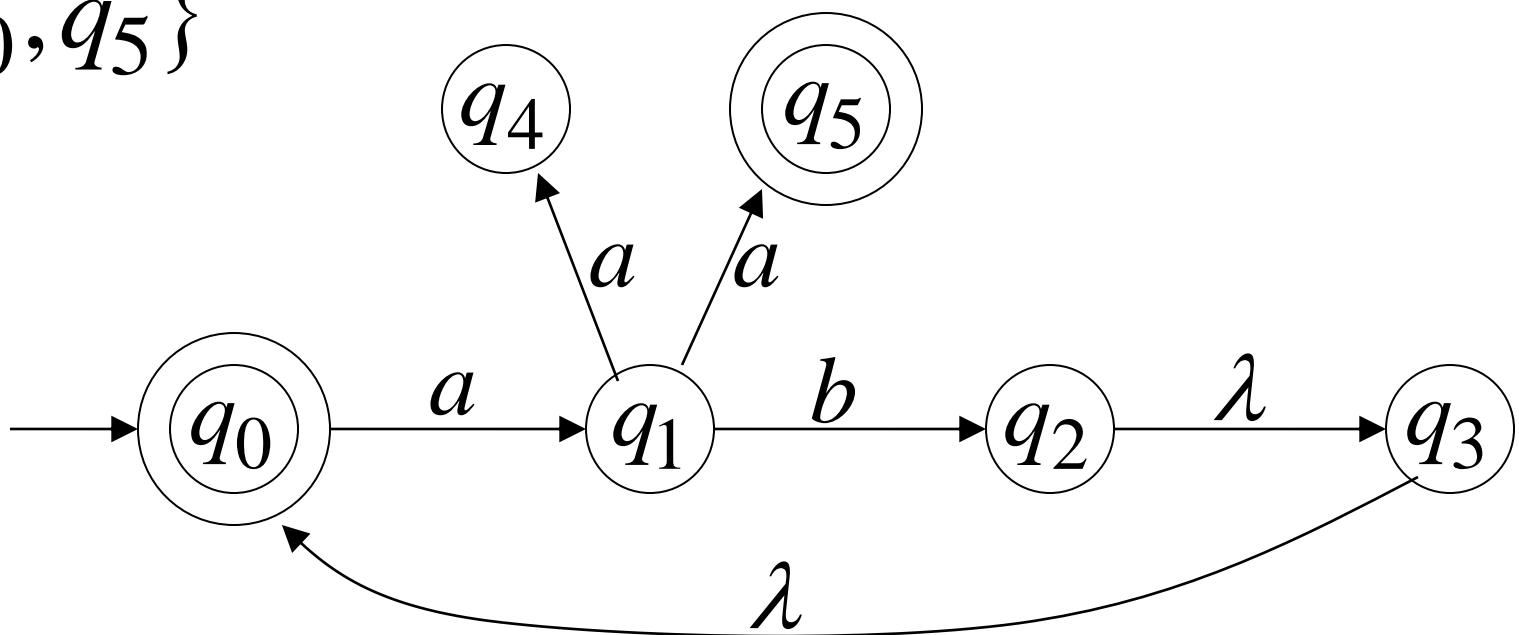
and there is some  $q_k \in F$  (accepting state)

$$w_m \in L(M)$$

$$\delta^*(q_0, w_m)$$



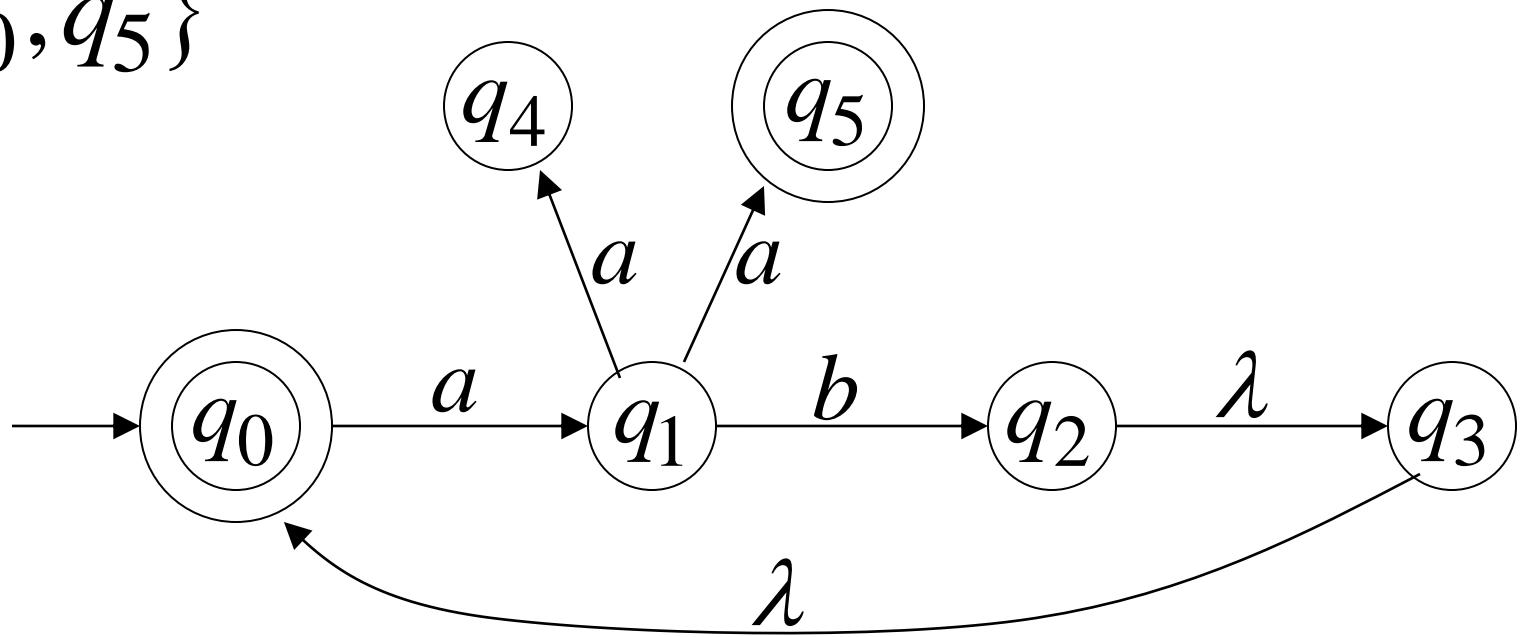
$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, aa) = \{q_4, \underline{q_5}\} \quad \xrightarrow{\hspace{1cm}} \quad aa \in L(M)$$

$\searrow \in F$

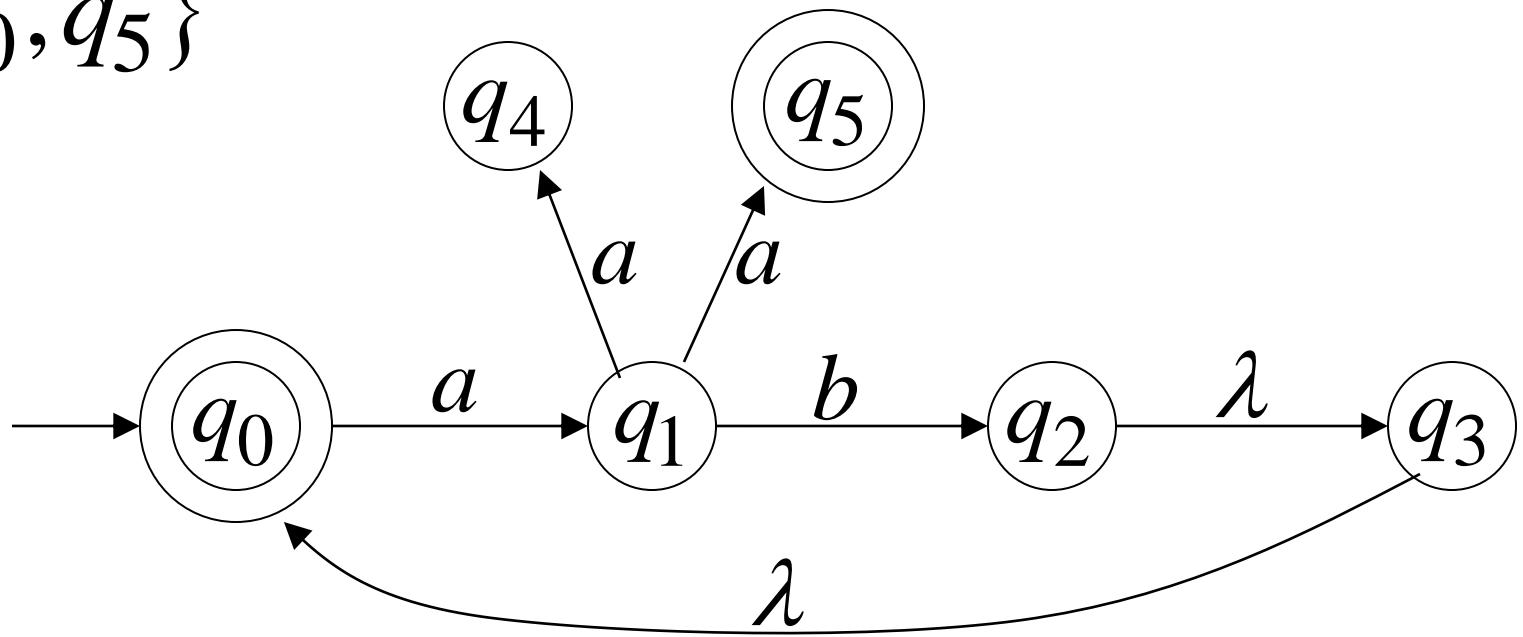
$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, ab) = \{q_2, q_3, \underline{q_0}\} \longrightarrow ab \in L(M)$$

$\searrow \in F$

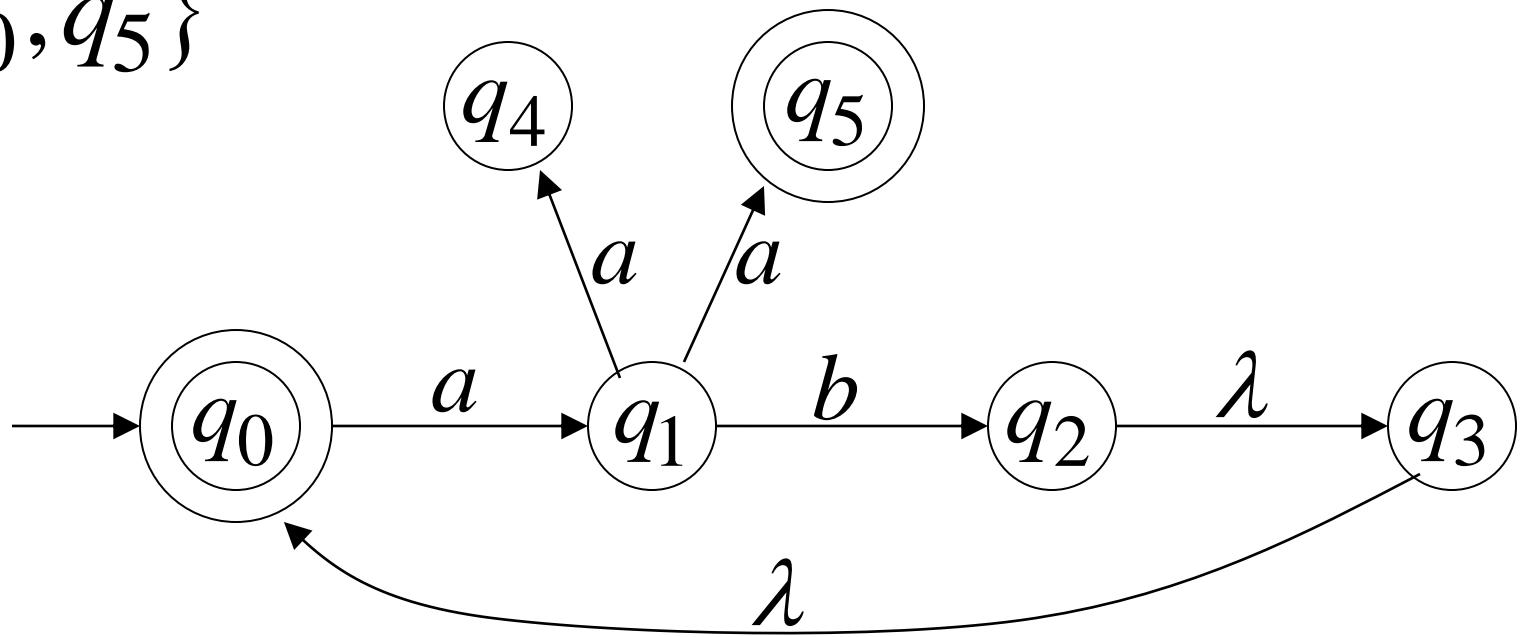
$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, aba) = \{q_4, \underline{q_5}\} \xrightarrow{\quad} aaba \in L(M)$$

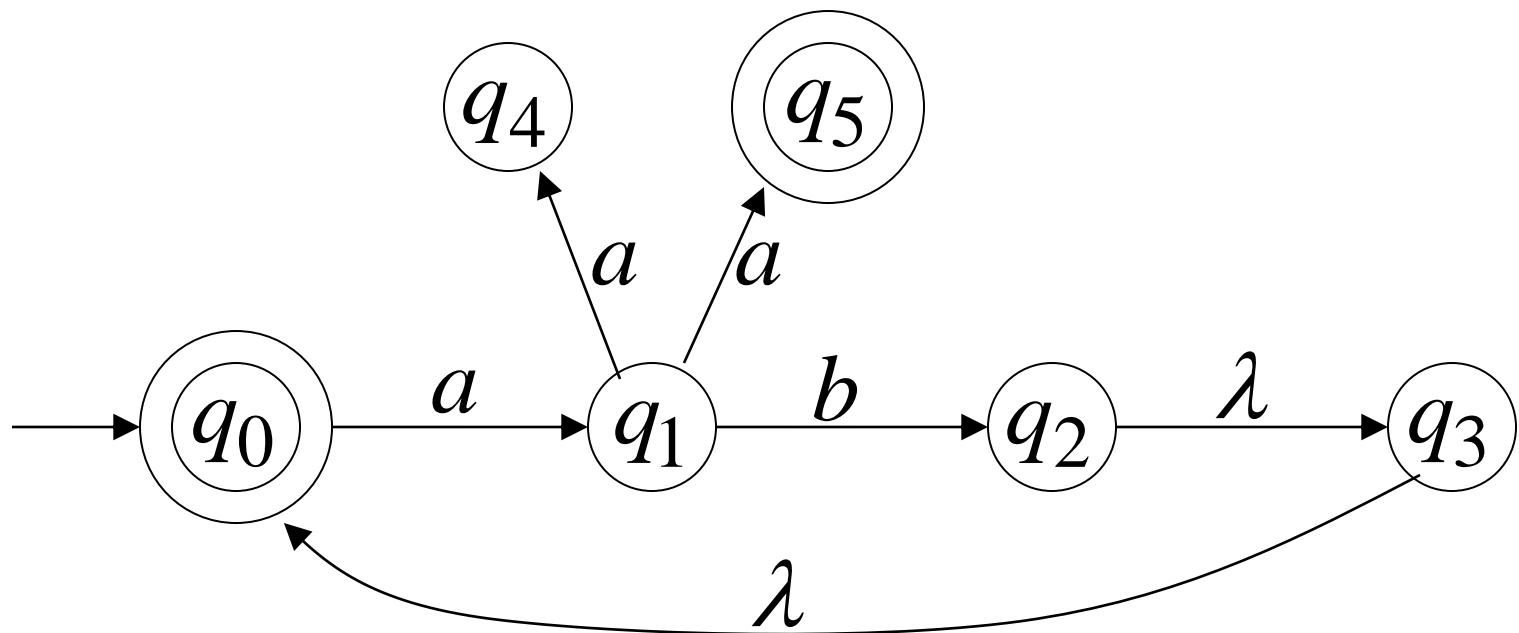
$\xrightarrow{\quad} \in F$

$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, aba) = \{q_1\} \quad \text{---} \longrightarrow \quad aba \notin L(M)$$

$\not\in F$

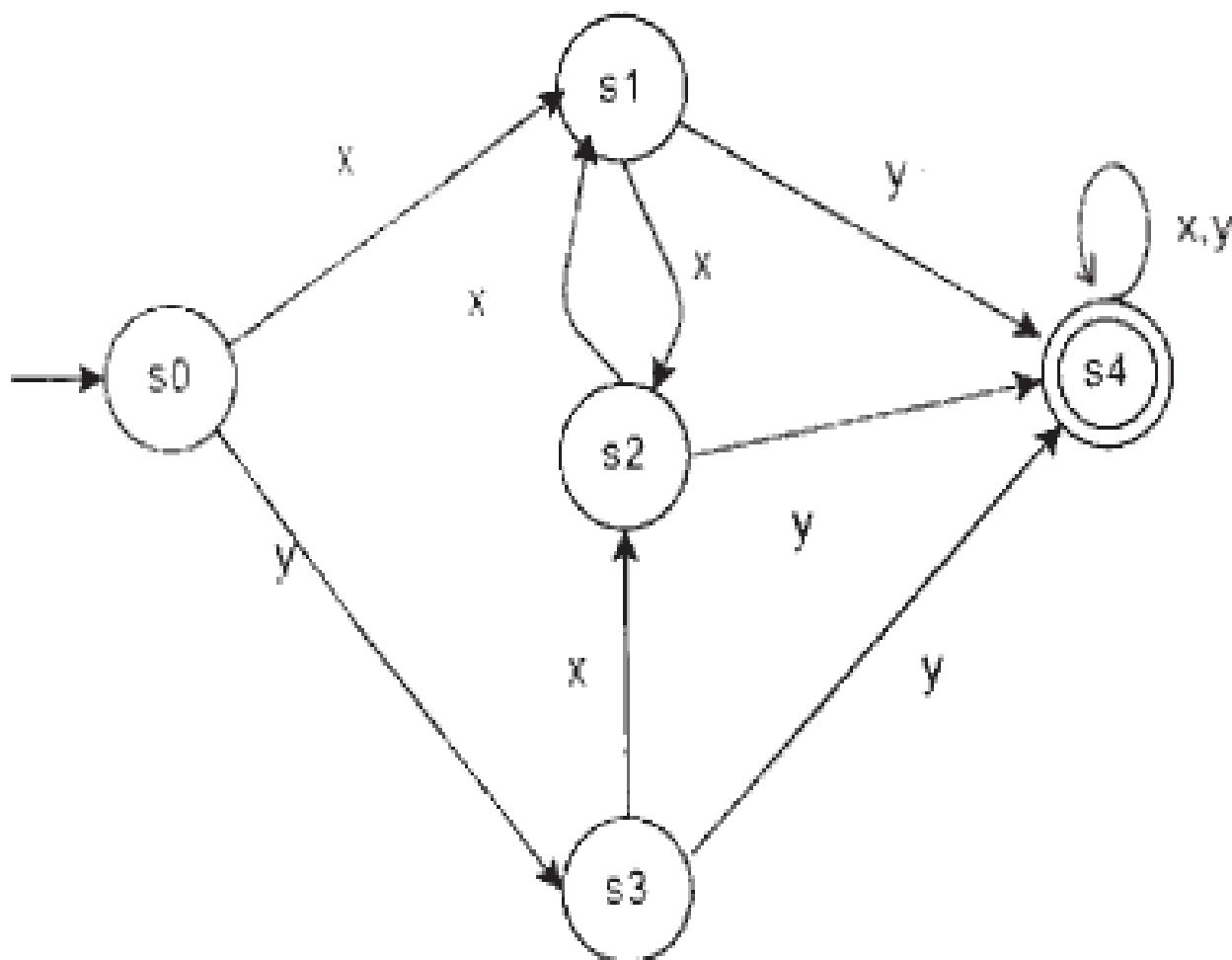


$$L(M) = \{ab\}^* \cup \{ab\}^* \{aa\}$$

# Recall

- NFA
- NFA-null
- Minimization of DFa

Minimize the following automata.



# THEORY OF COMPUTATION

---

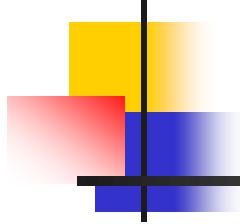
Unit I

## FINITE STATE MACHINES

---

**Jagadish Kashinath Kamble** (M.Tech, IIT Kharagpur)

Dept. of Information Technology,  
Pune Institute of Computer Technology, Pune



# Recall

---

- NFA
- NFA with null/Epsilon Transition

# Equivalence of Machines

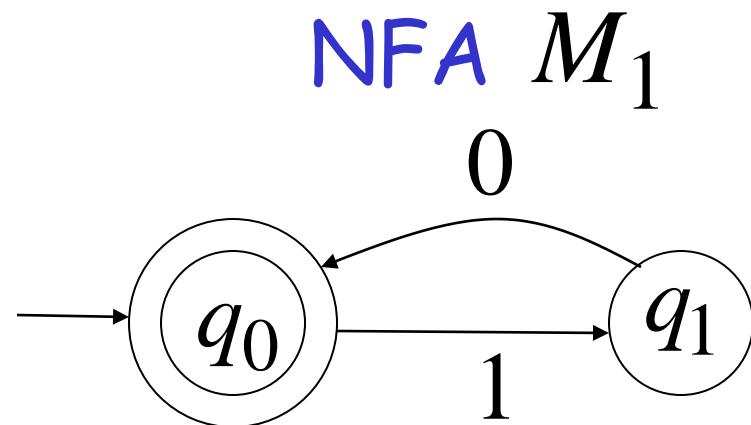
Definition:

Machine  $M_1$  is equivalent to machine  $M_2$

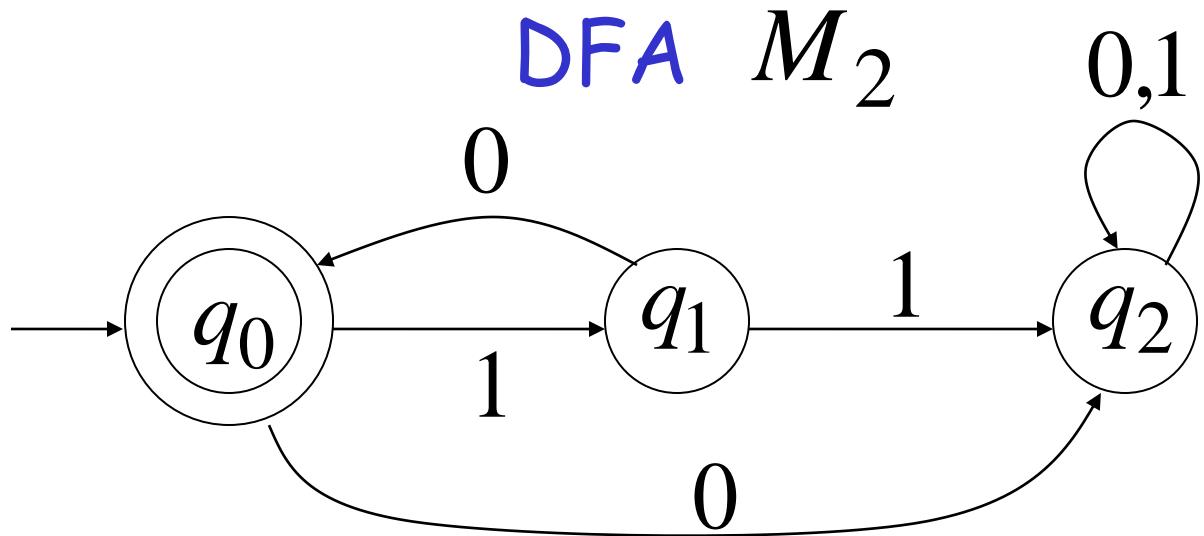
if  $L(M_1) = L(M_2)$

# Example of equivalent machines

$$L(M_1) = \{10\}^*$$



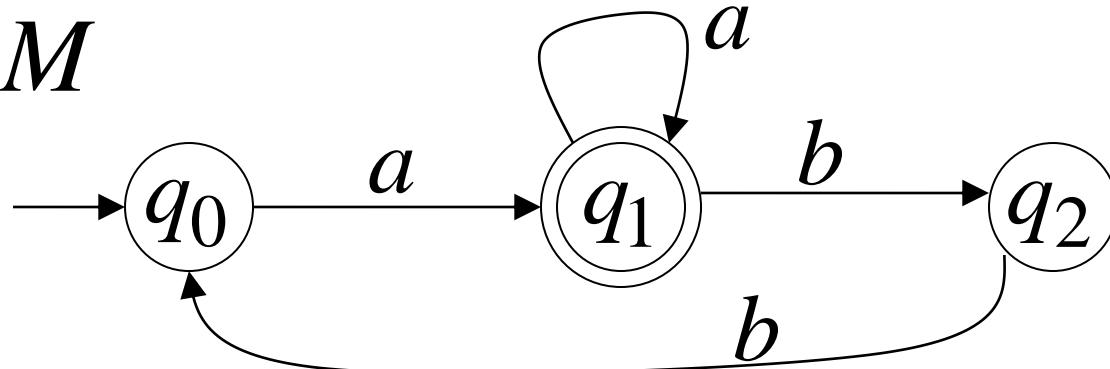
$$L(M_2) = \{10\}^*$$



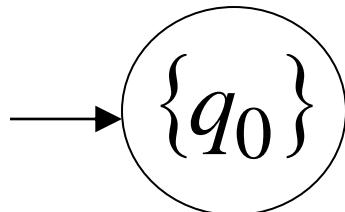
NFA to DFA

# Conversion NFA to DFA

NFA  $M$

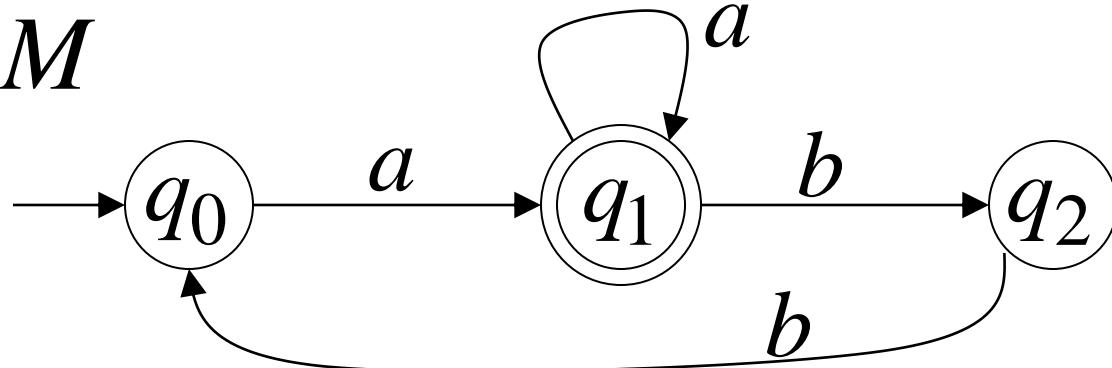


DFA  $M'$

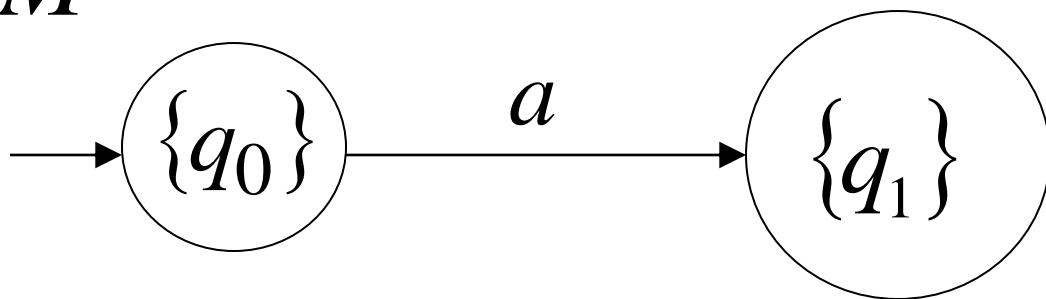


$$\delta^*(q_0, a) = \{q_1\}$$

NFA  $M$

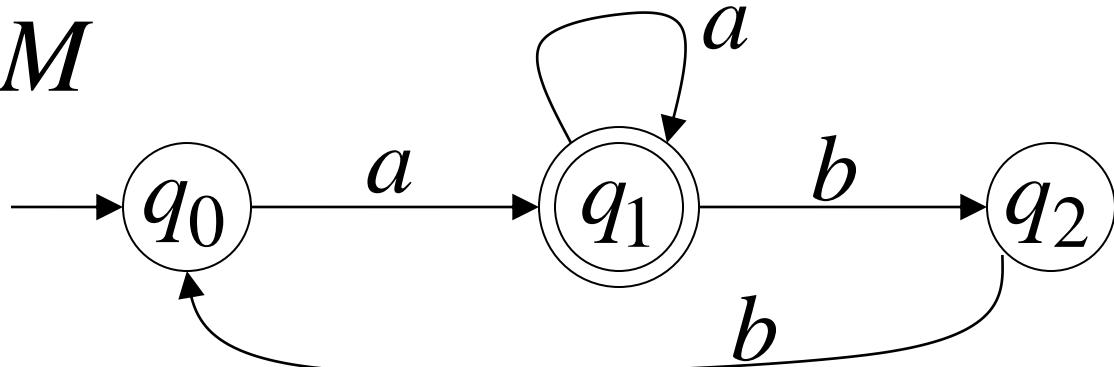


DFA  $M'$

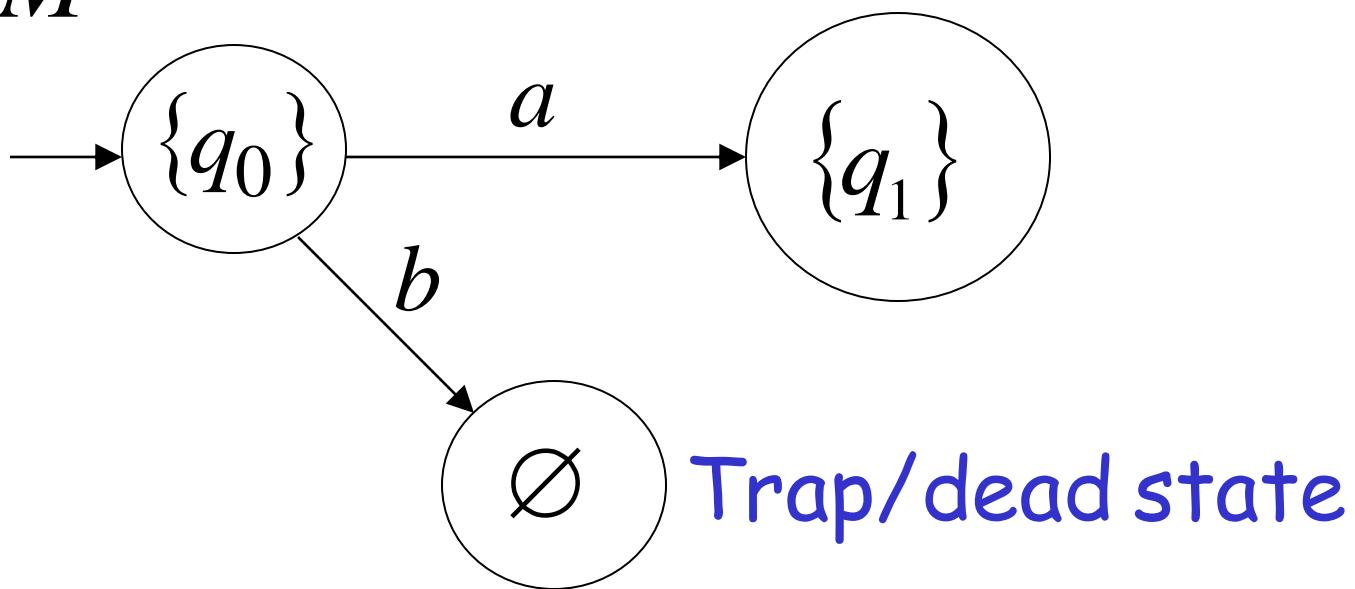


$$\delta^*(q_0, b) = \emptyset \quad \text{empty set}$$

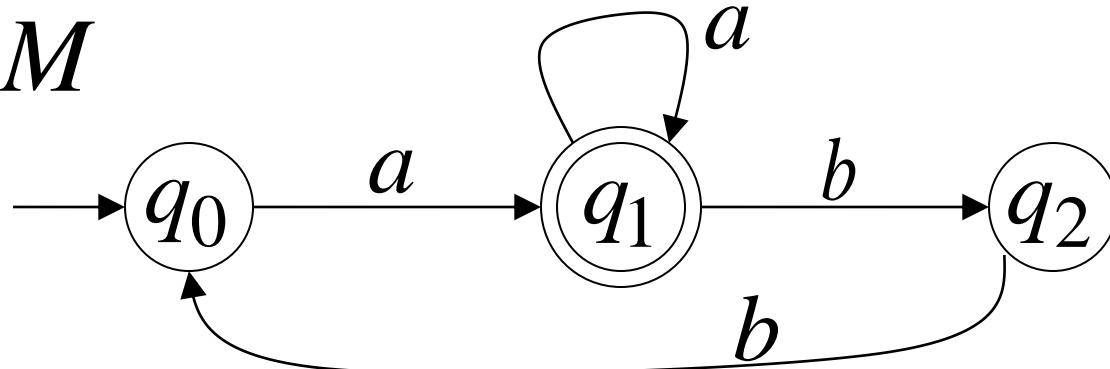
NFA  $M$



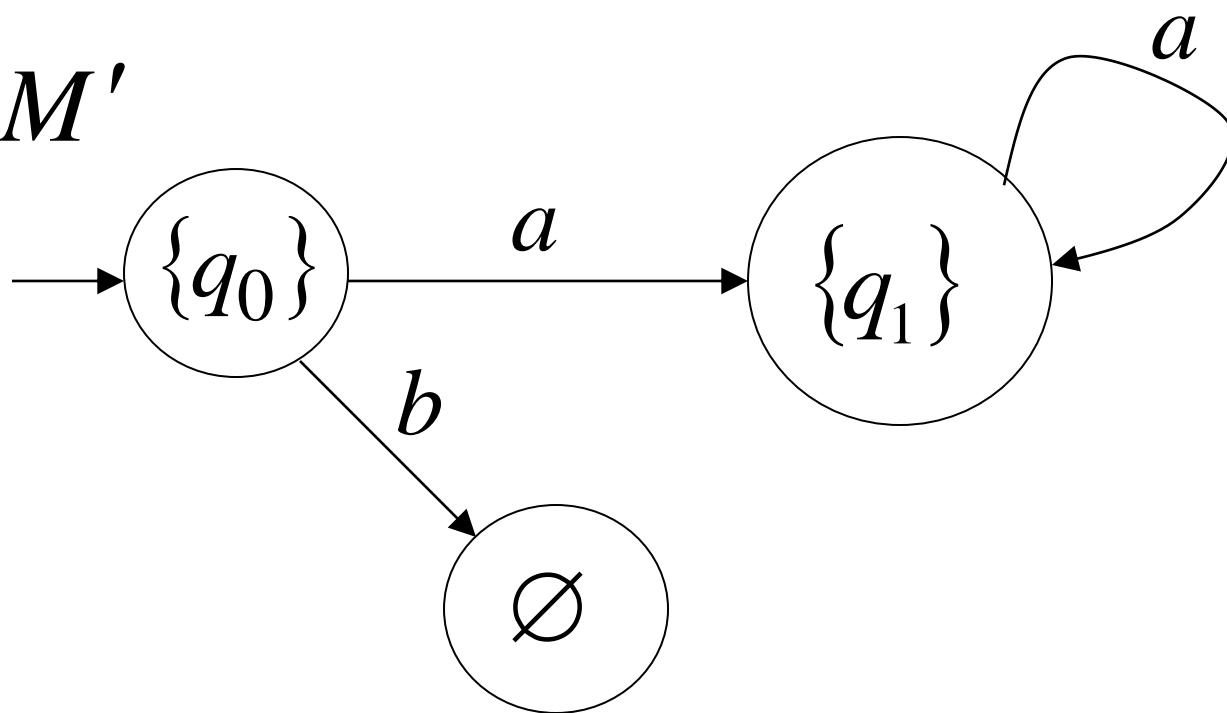
DFA  $M'$



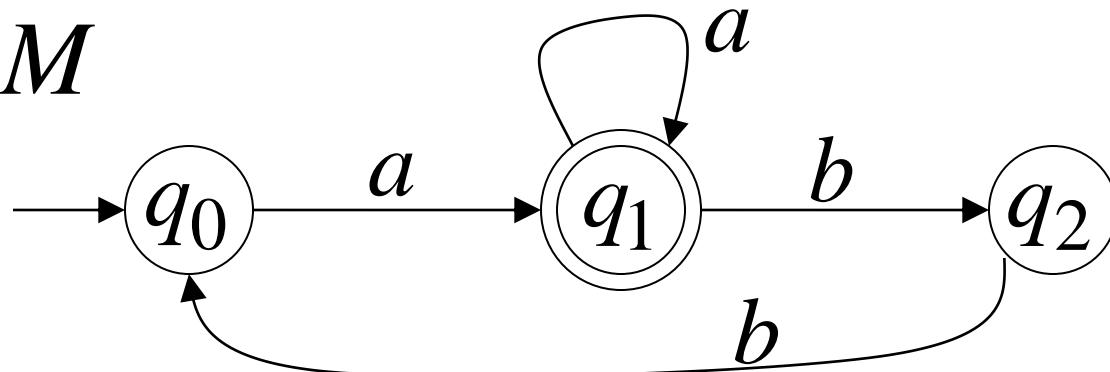
**NFA**  $M$



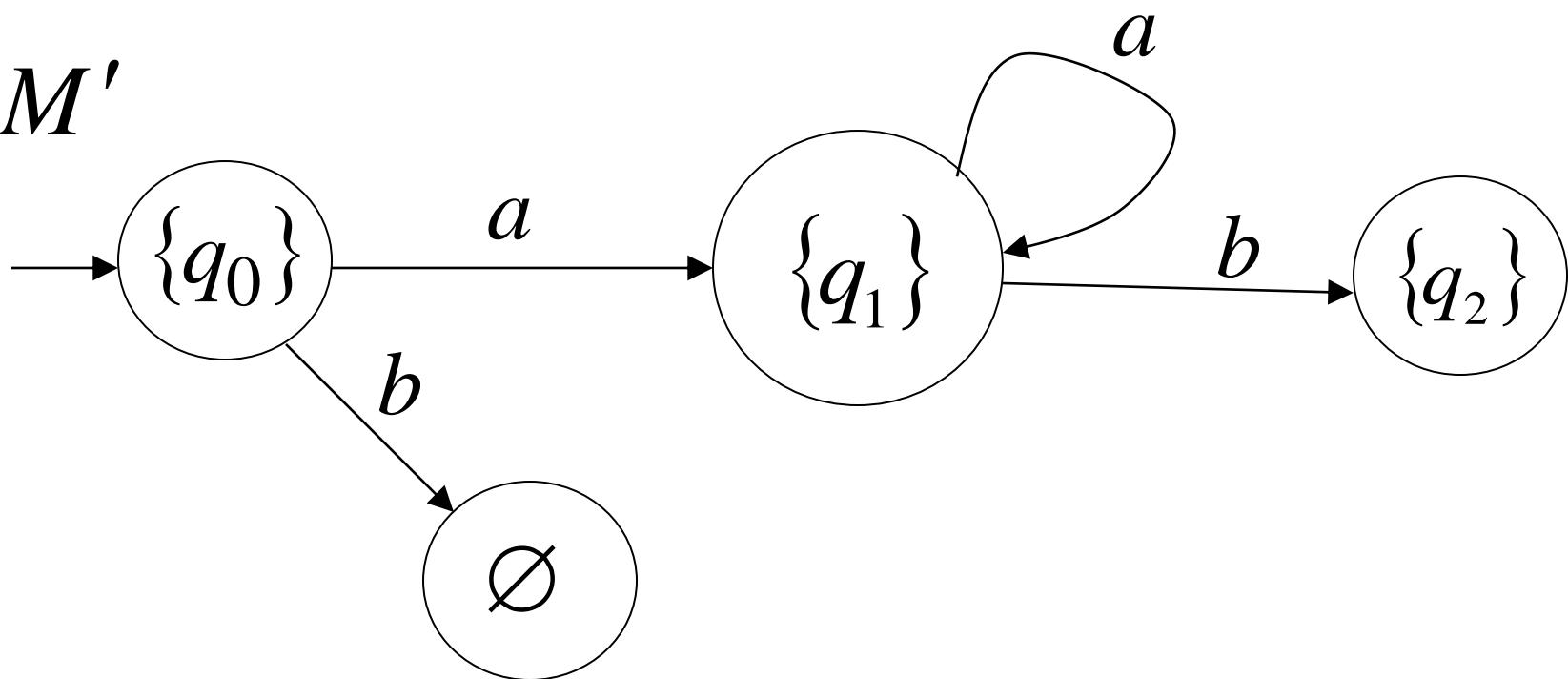
**DFA**  $M'$



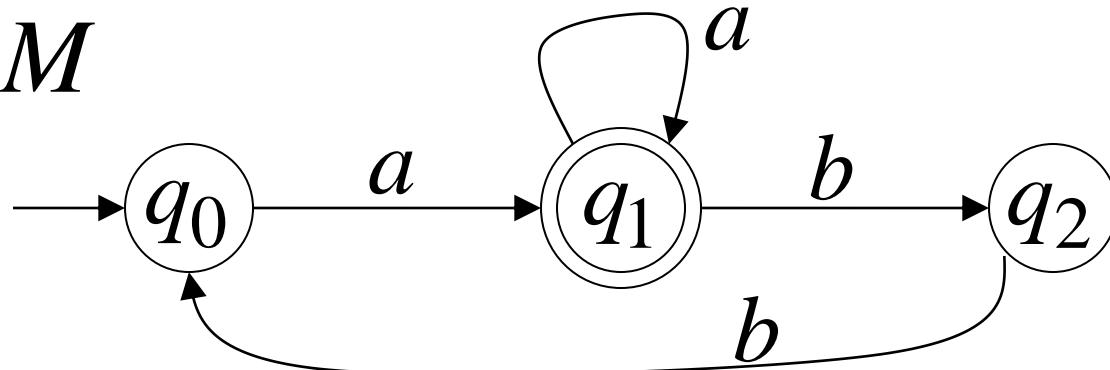
**NFA**  $M$



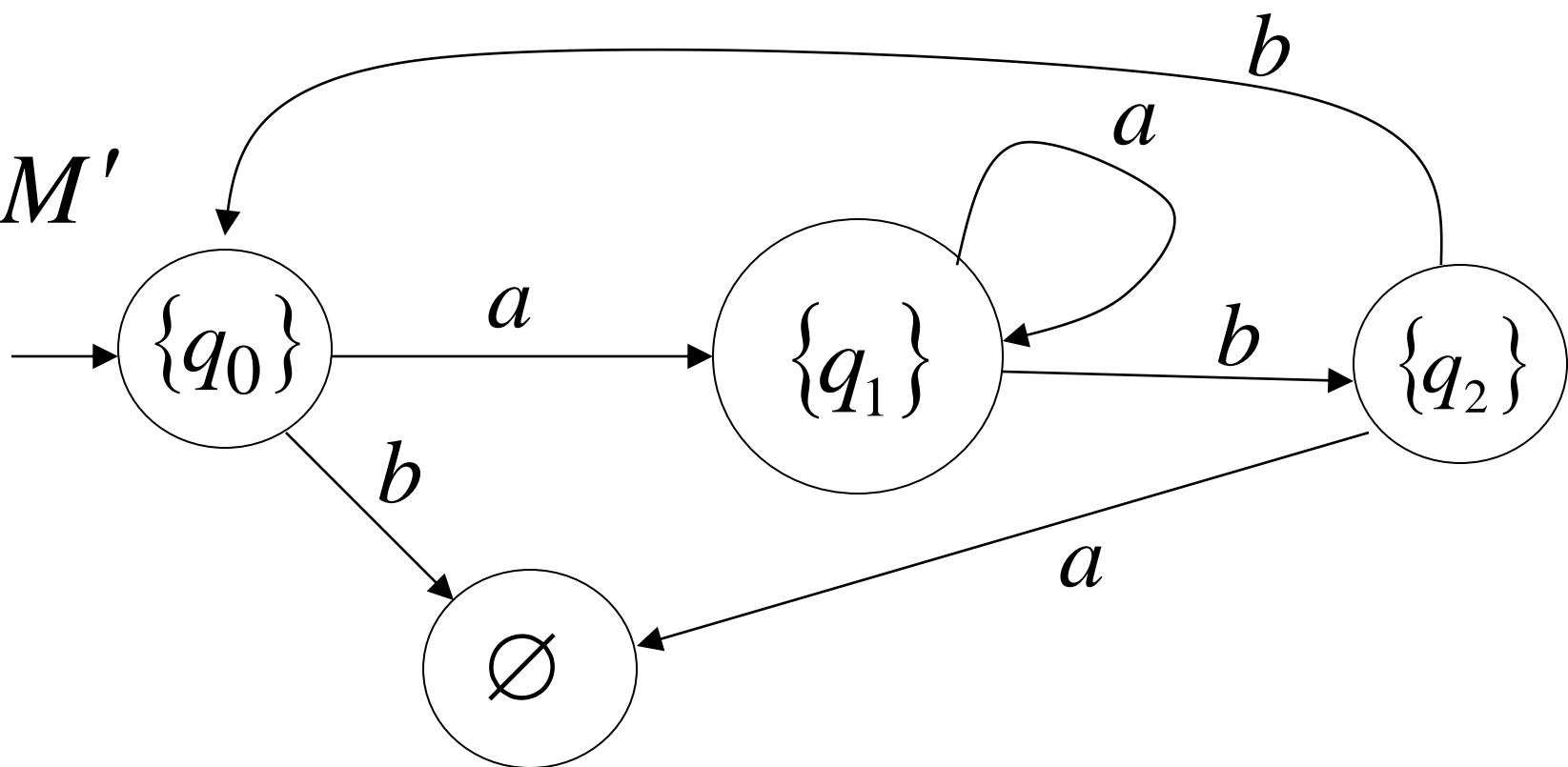
**DFA**  $M'$



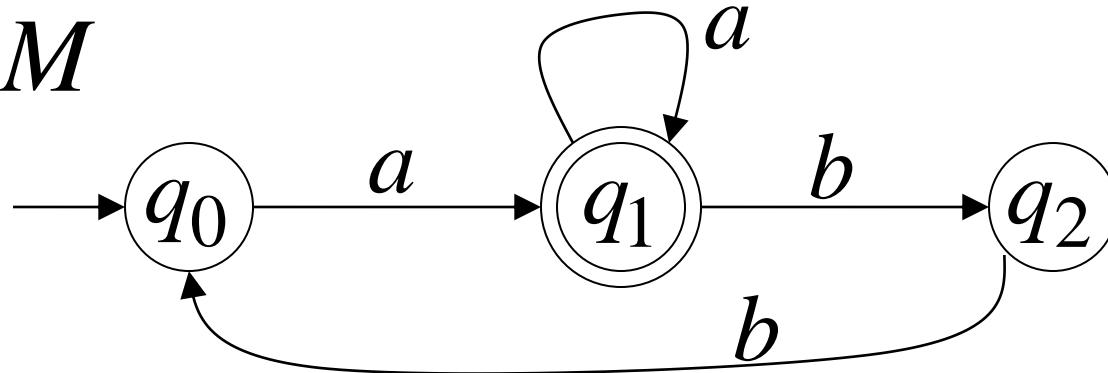
NFA  $M$



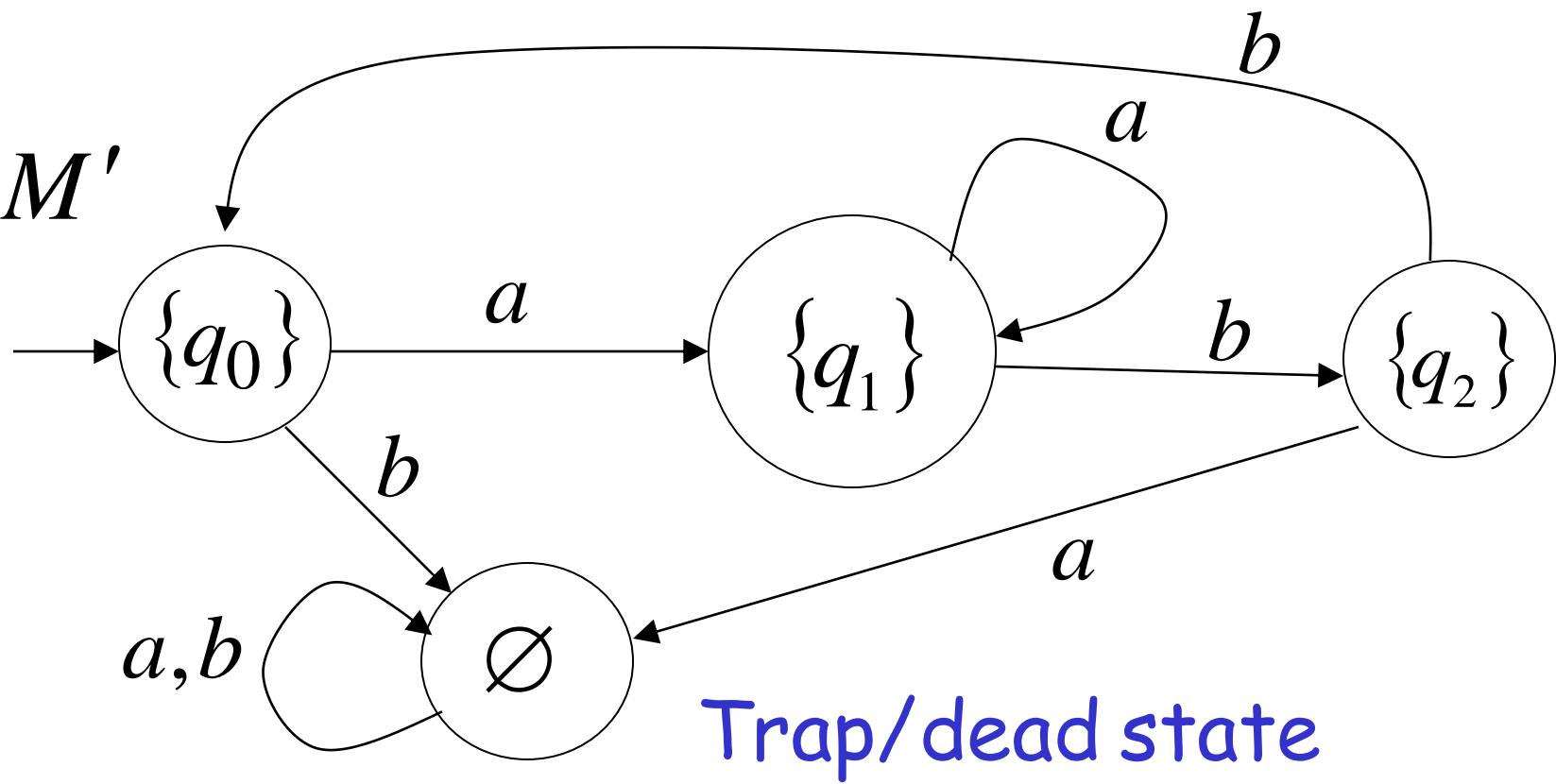
DFA  $M'$



NFA  $M$

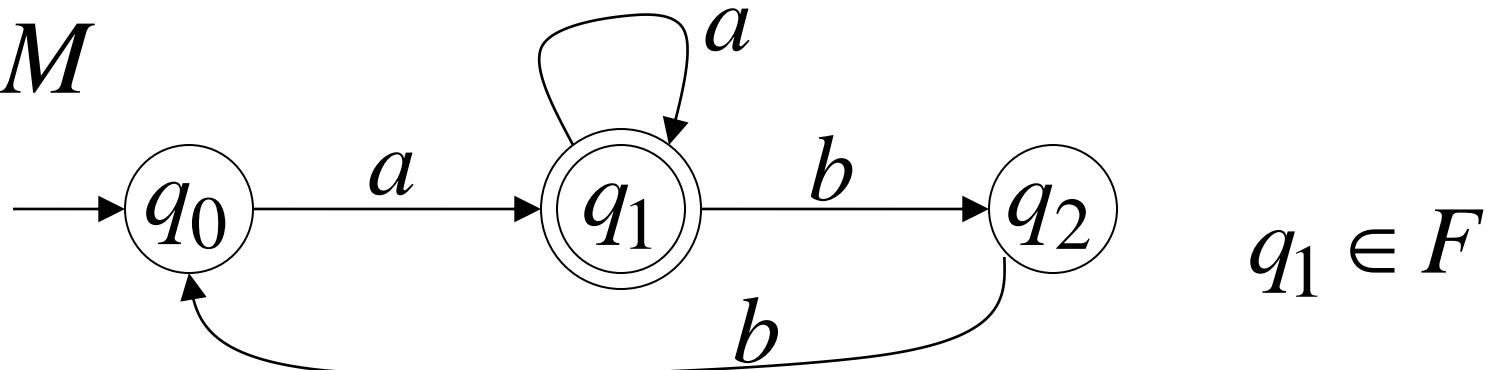


DFA  $M'$



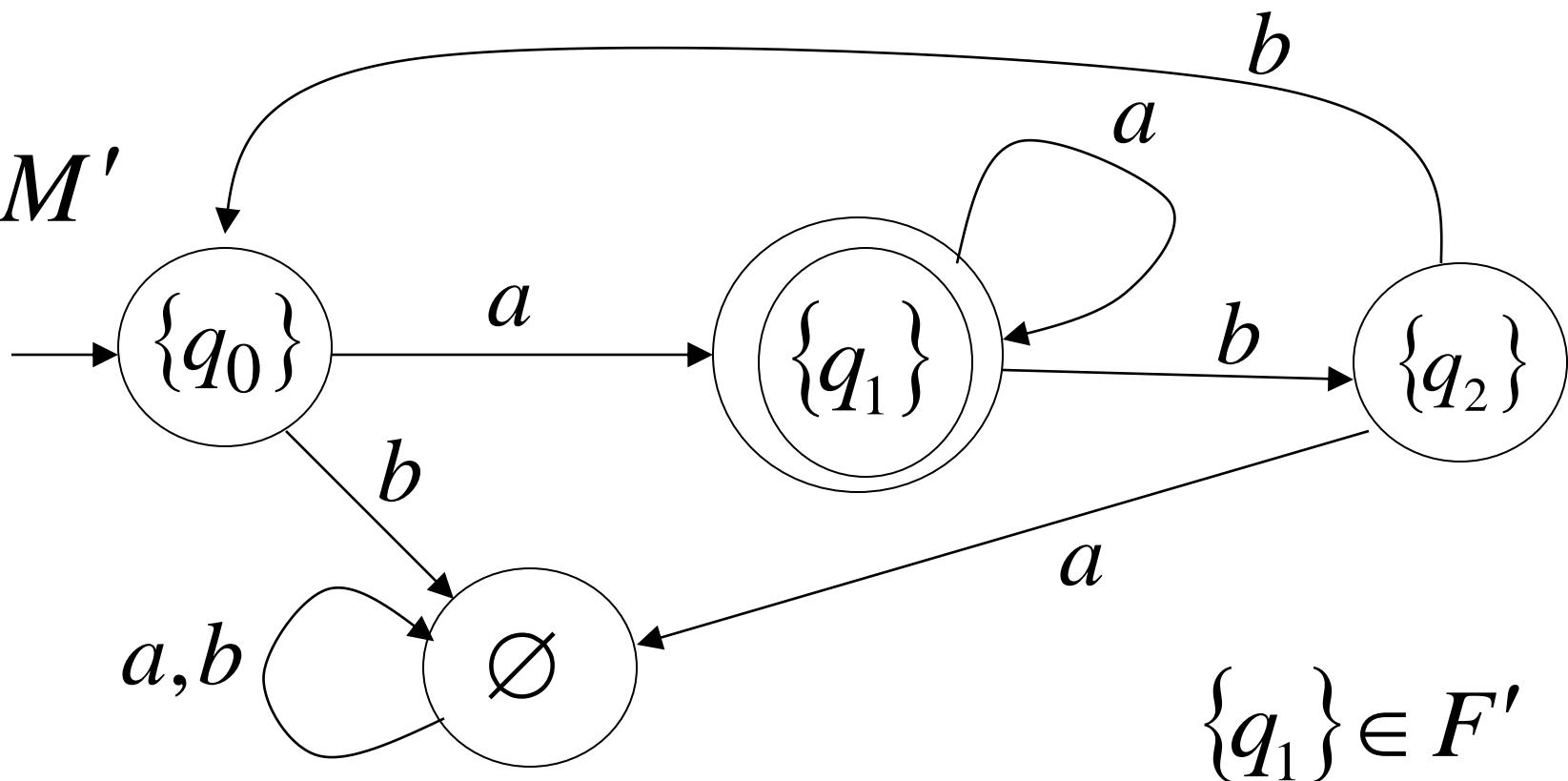
# END OF CONSTRUCTION

NFA  $M$



$$q_1 \in F$$

DFA  $M'$



$$\{q_1\} \in F'$$

# Conversion NFA to DFA

# General Conversion Procedure

Input: an NFA  $M$

Output: an equivalent DFA  $M'$   
with  $L(M) = L(M')$

The NFA has states  $q_0, q_1, q_2, \dots$

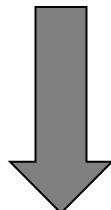
The DFA has states from the power set

$\emptyset, \{q_0\}, \{q_1\}, \{q_0, q_1\}, \{q_1, q_2, q_3\}, \dots$

# Conversion Procedure Steps

step

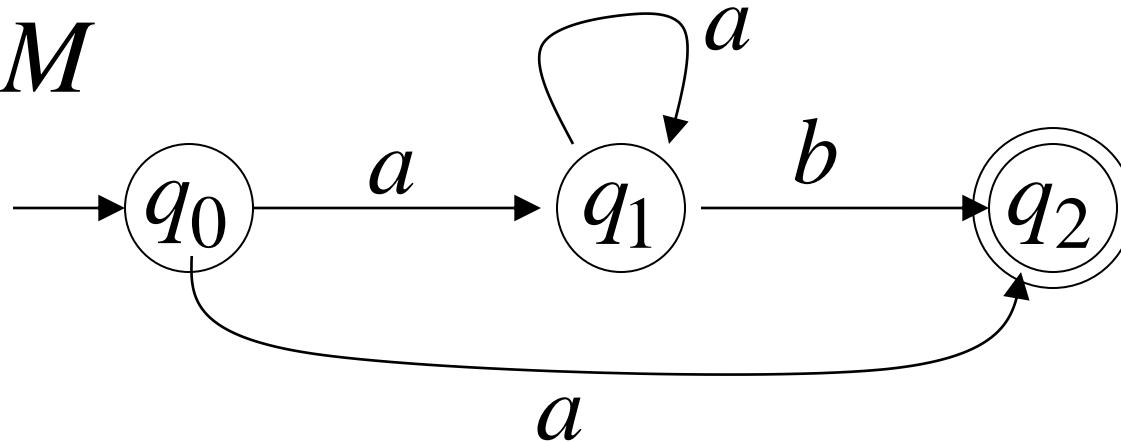
1. Initial state of NFA:  $q_0$



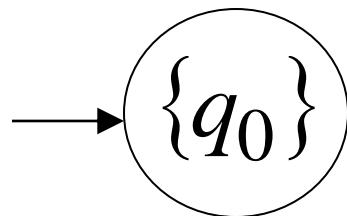
Initial state of DFA:  $\{q_0\}$

# Example

NFA  $M$

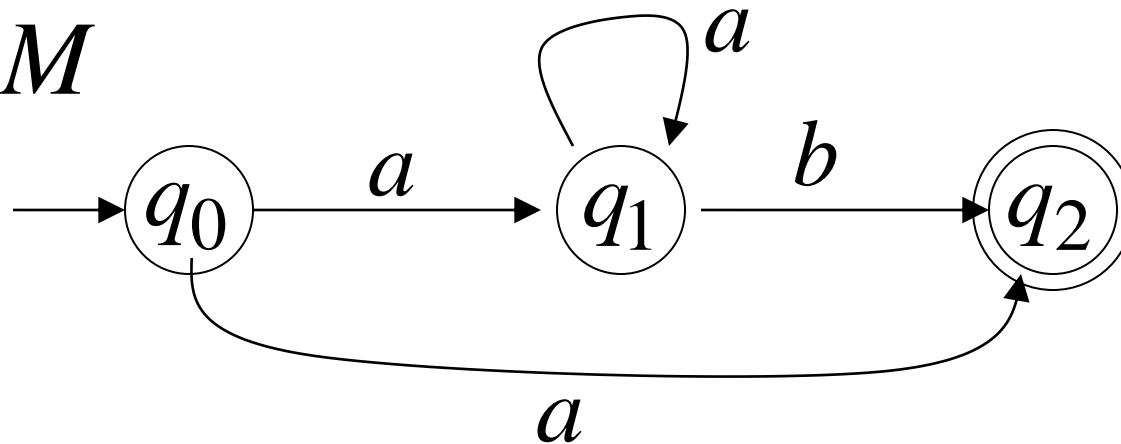


DFA  $M'$

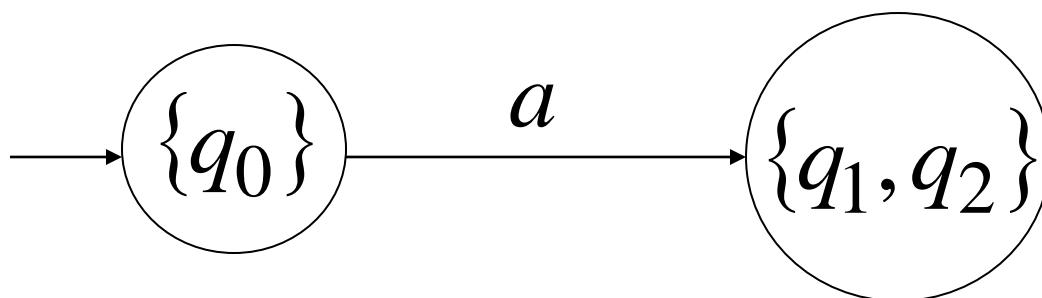


# Example

NFA  $M$

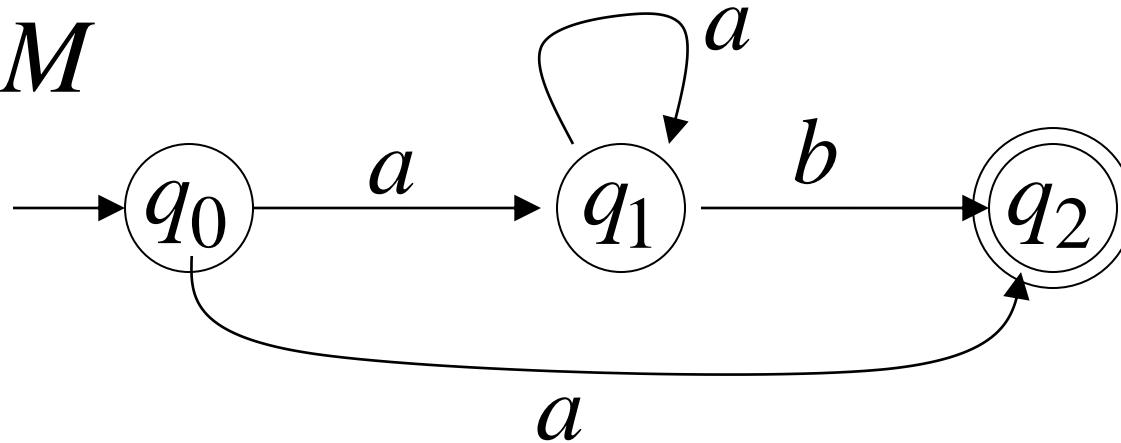


DFA  $M'$      $\delta(\{q_0\}, a) = \{q_1, q_2\}$

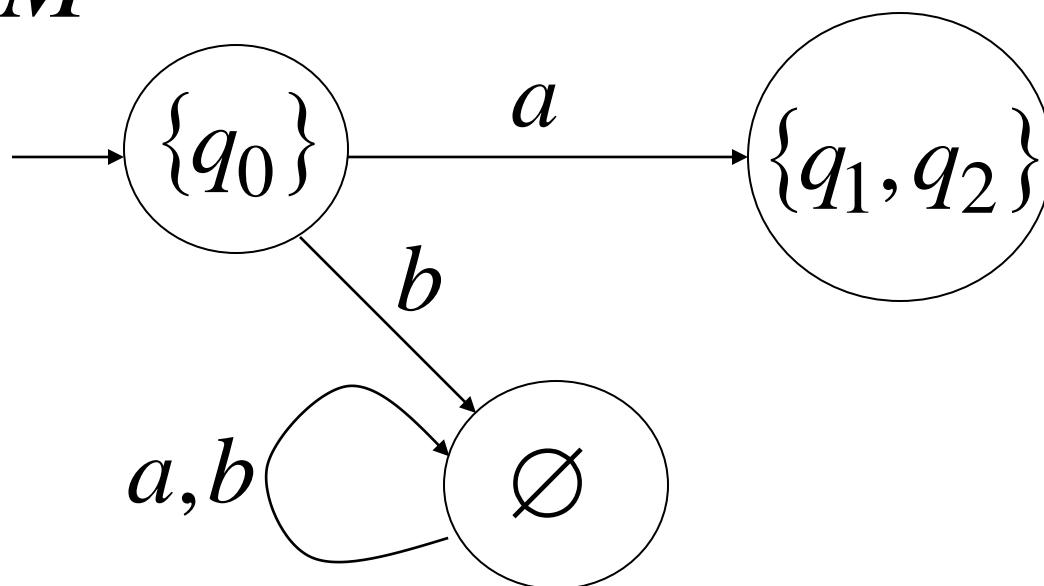


# Example

NFA  $M$



DFA  $M'$



step

2. For every DFA's state  $\{q_i, q_j, \dots, q_m\}$

compute in the NFA

$$\left. \begin{array}{l} \delta^*(q_i, a) \\ \cup \delta^*(q_j, a) \\ \dots \\ \cup \delta^*(q_m, a) \end{array} \right\} = \{q'_k, q'_l, \dots, q'_n\}$$

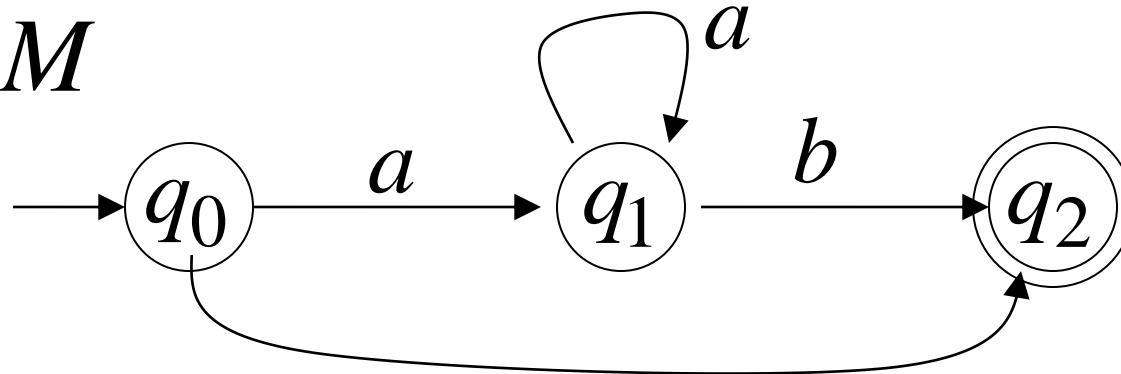
Union

add transition to DFA

$$\delta(\{q_i, q_j, \dots, q_m\}, a) = \{q'_k, q'_l, \dots, q'_n\}$$

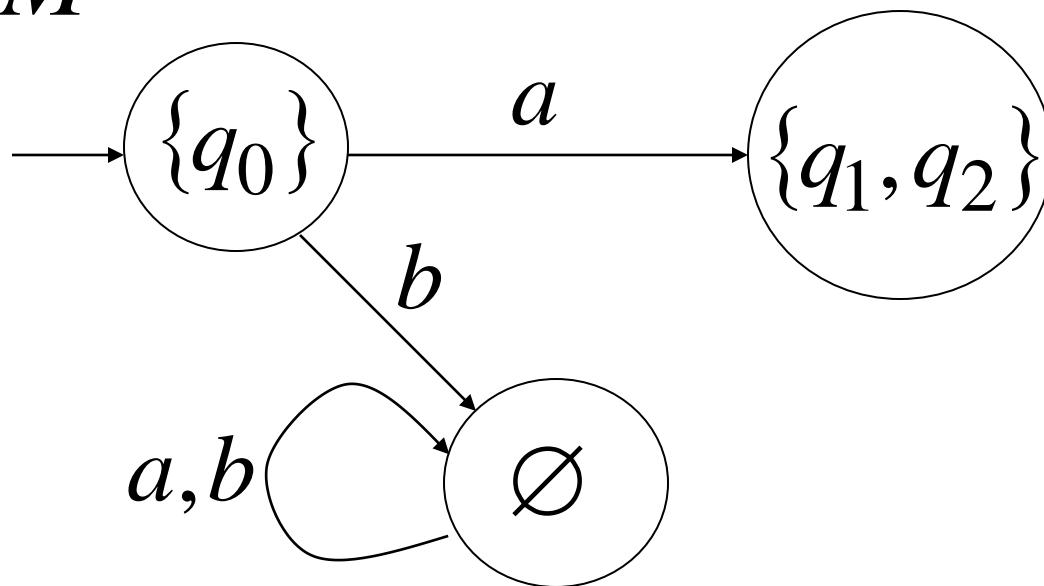
# Example

NFA  $M$



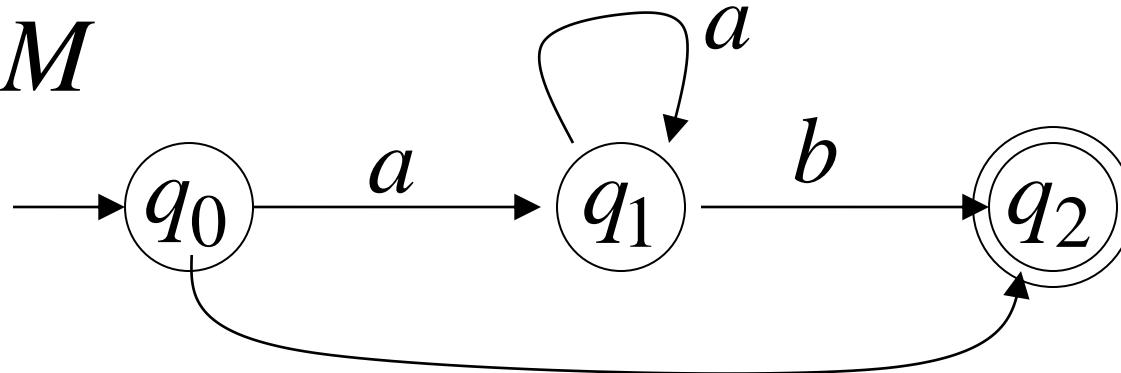
$$\delta^*(\{q_1, q_2\}, a) = \delta^*(q_1, a) \cup \delta^*(q_2, a) = \{q_1\} \cup \Theta = \{q_1\}$$

DFA  $M'$



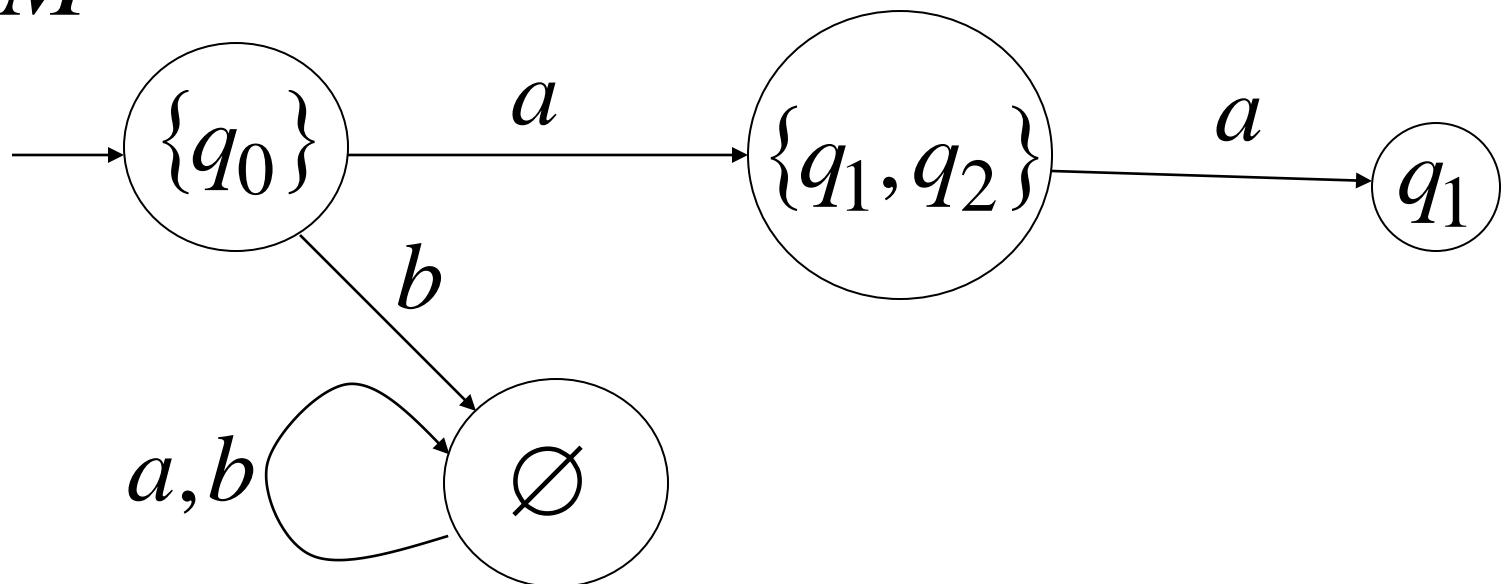
# Example

NFA  $M$



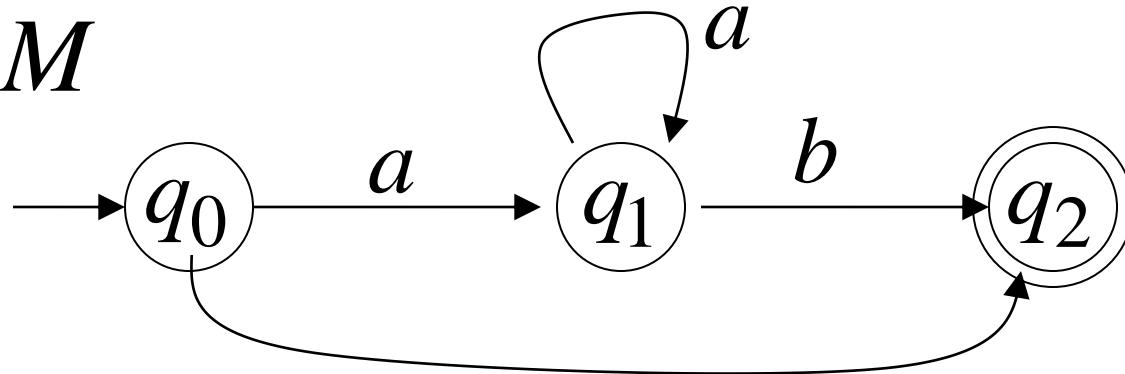
$$\delta^*(\{q_1, q_2\}, a) = \delta^*(q_1, a) \cup \delta^*(q_2, a) = \{q_1\} \cup \Theta = \{q_1\}$$

DFA  $M'$



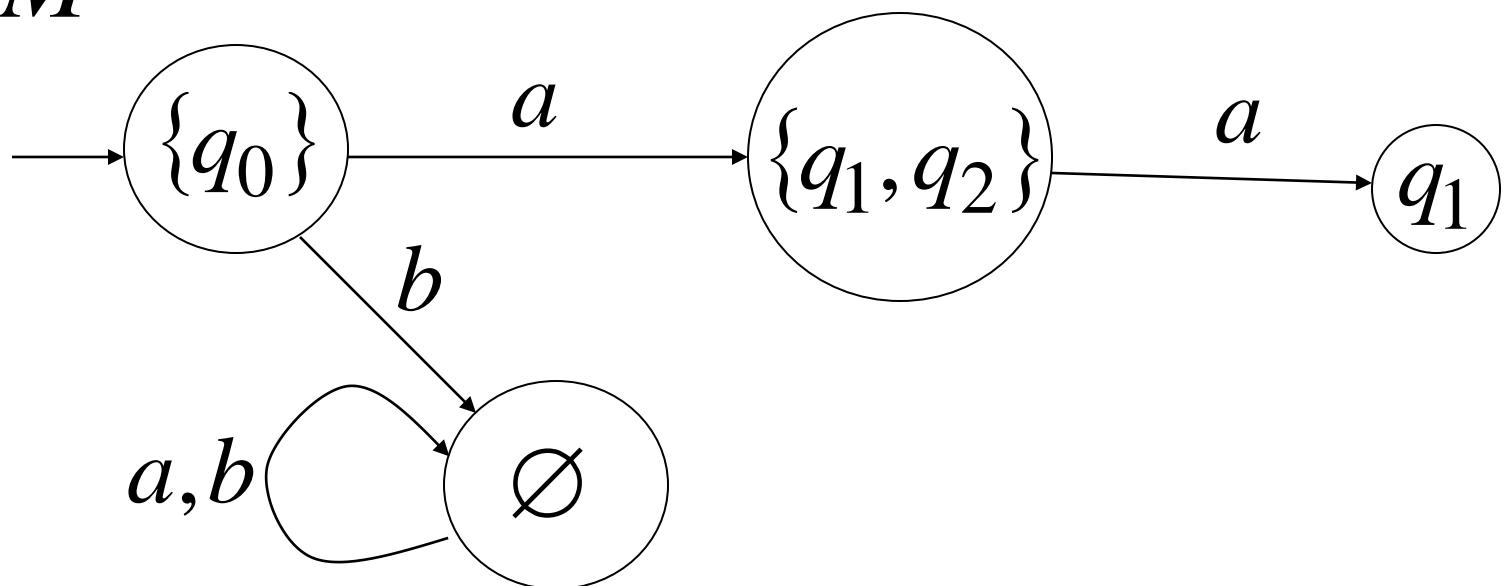
# Example

NFA  $M$



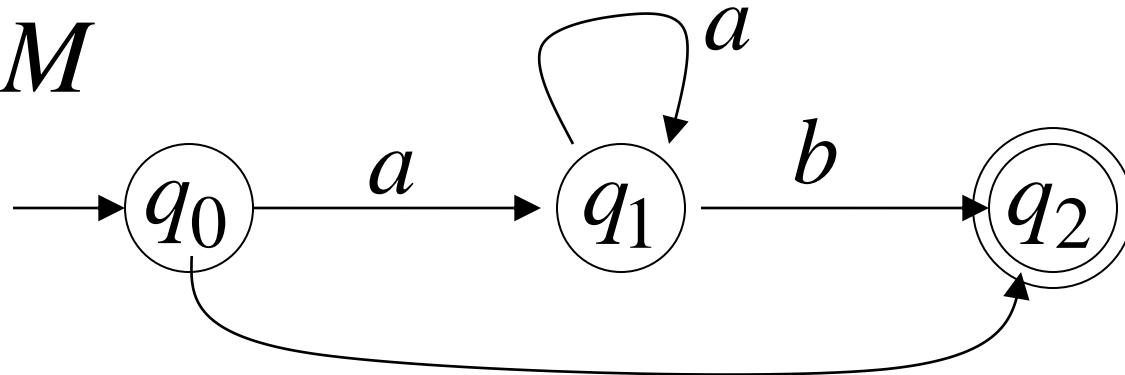
$$\delta^*(\{q_1, q_2\}, b) = \delta^*(q_1, b) \cup \delta^*(q_2, b) = \{q_2\} \cup \Theta = \{q_2\}$$

DFA  $M'$



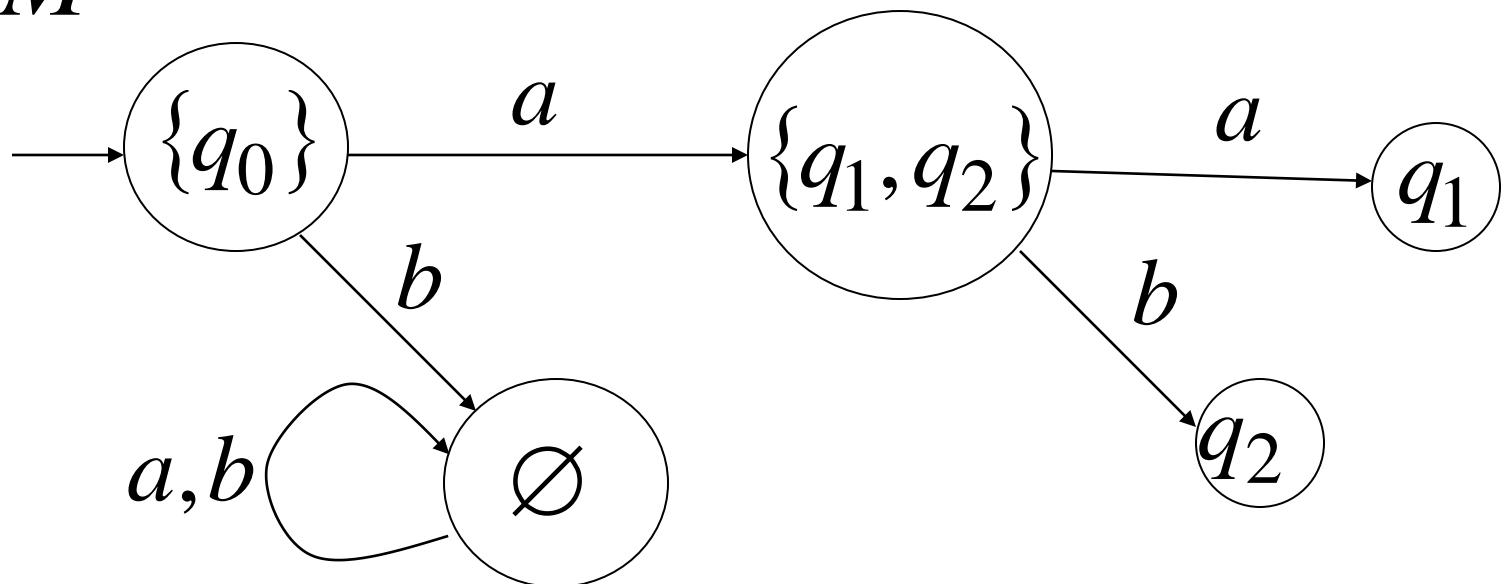
# Example

NFA  $M$



$$\delta^*(\{q_1, q_2\}, b) = \delta^*(q_1, b) \cup \delta^*(q_2, b) = \{q_2\} \cup \Theta = \{q_2\}$$

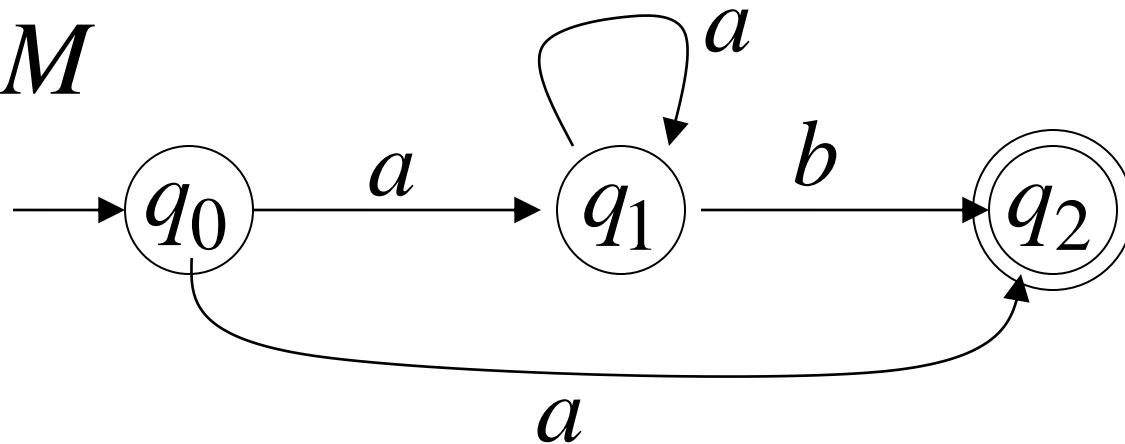
DFA  $M'$



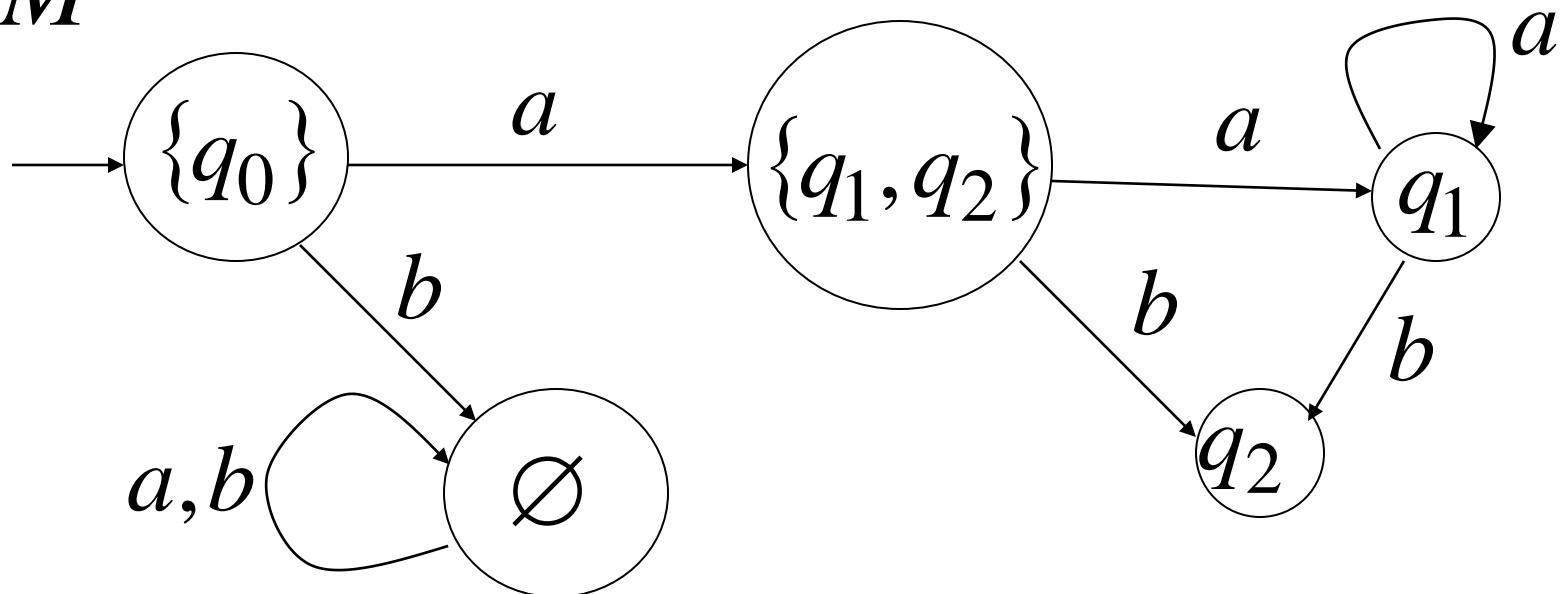
**step 3.** Repeat Step 2 for every state in DFA  
and symbols in alphabet until no more  
states can be added in the DFA

# Example

NFA  $M$

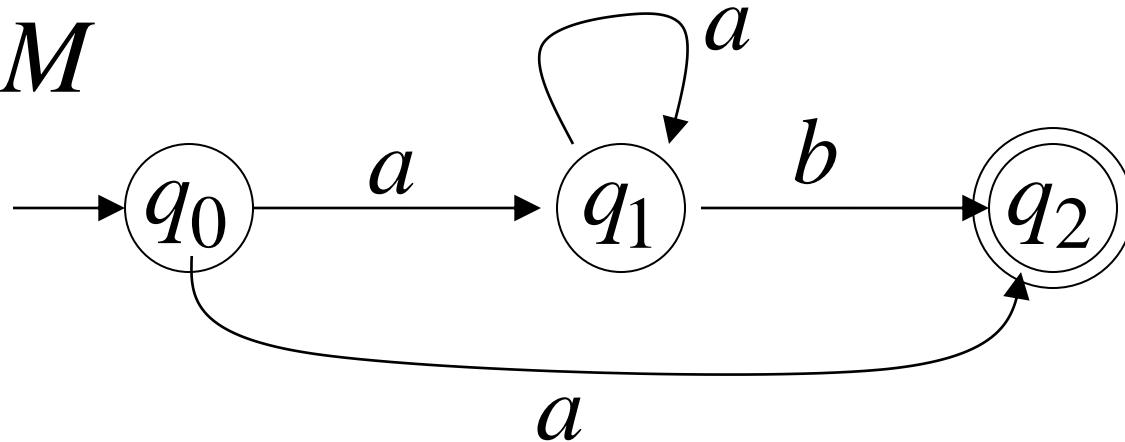


DFA  $M'$

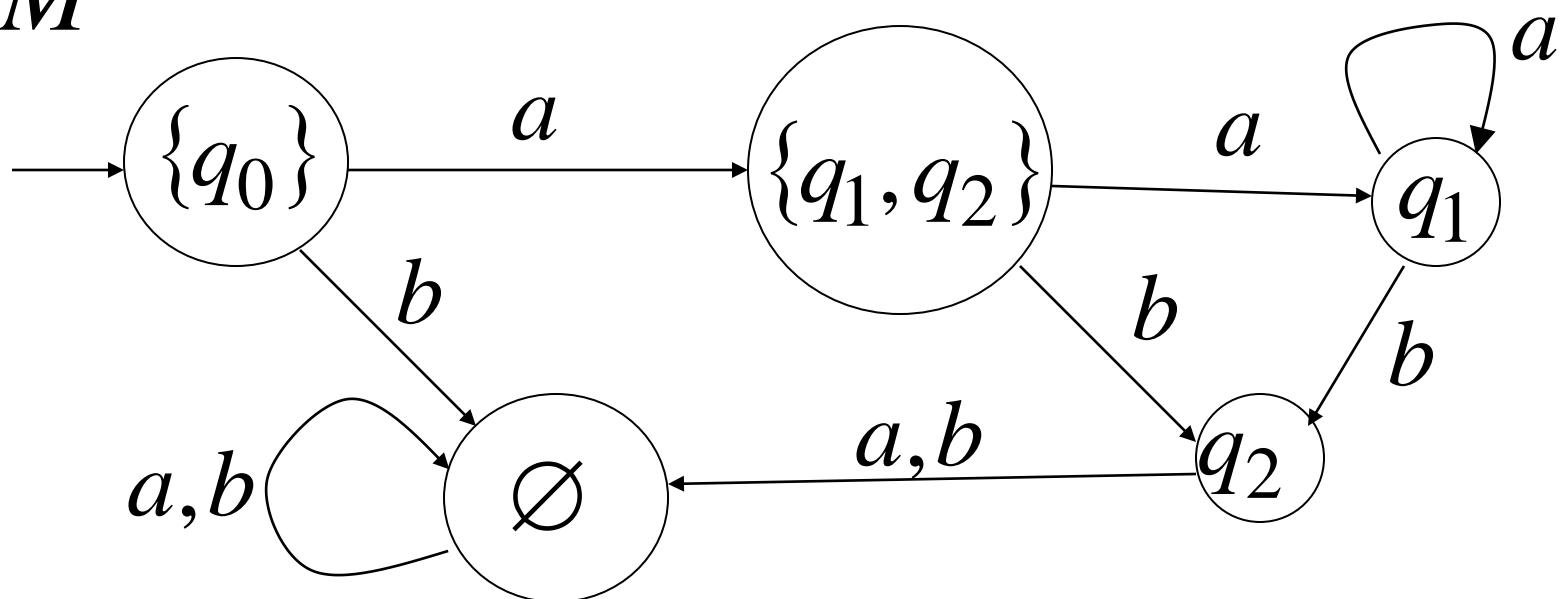


# Example

NFA  $M$



DFA  $M'$



step

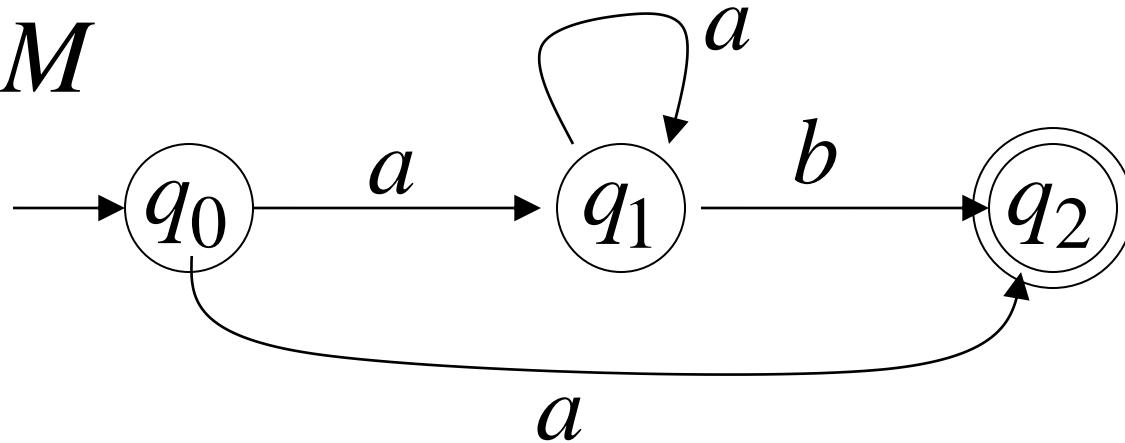
4. For any DFA state  $\{q_i, q_j, \dots, q_m\}$

if some  $q_j$  is accepting state in NFA

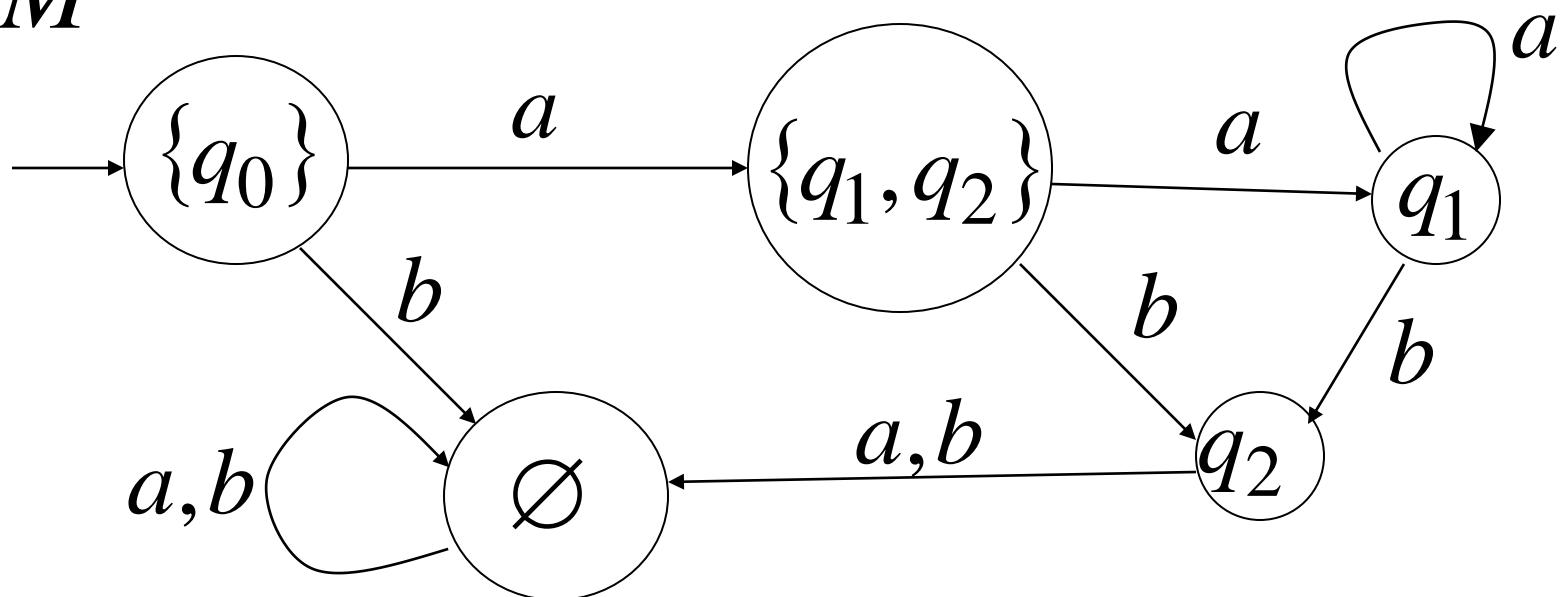
Then,  $\{q_i, q_j, \dots, q_m\}$   
is accepting state in DFA

# Example

NFA  $M$

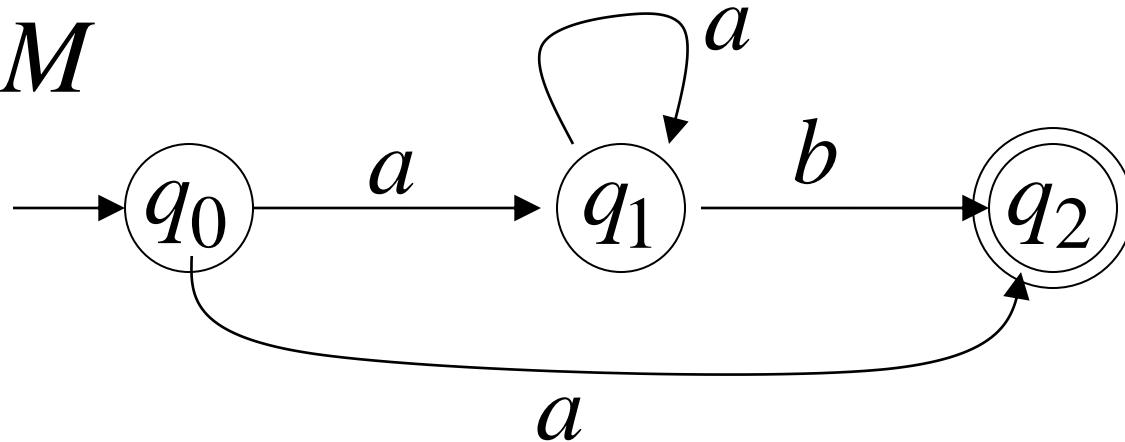


DFA  $M'$



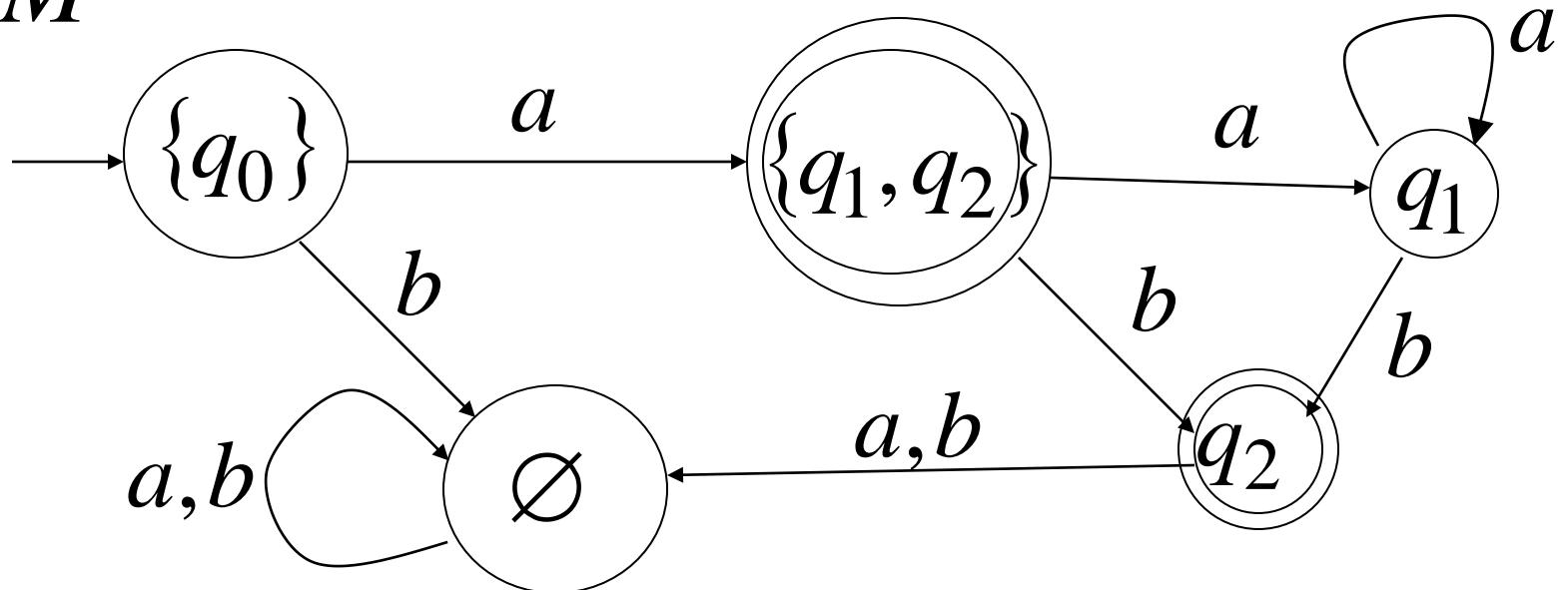
# Example

NFA  $M$



$$q_2 \in F$$

DFA  $M'$

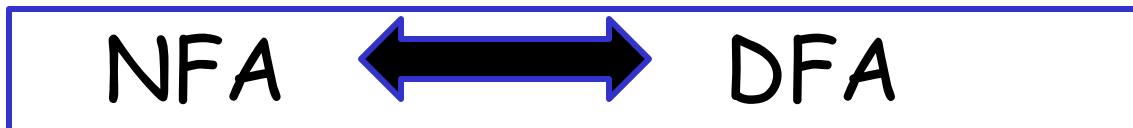
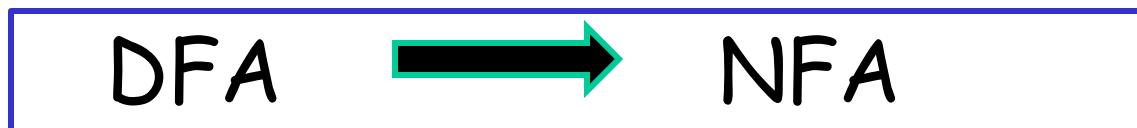
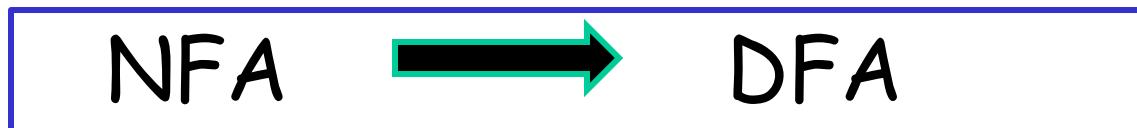


$$\{q_1, q_2\} \in F'$$

## Lemma:

If we convert NFA  $M$  to DFA  $M'$   
then the two automata are equivalent:

$$L(M) = L(M')$$



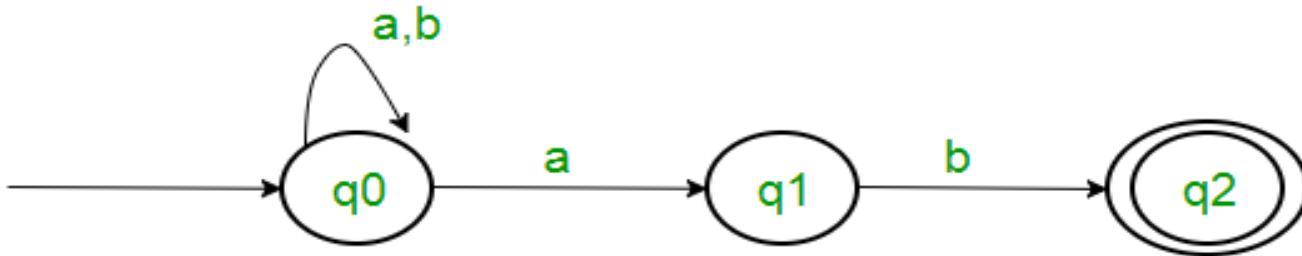
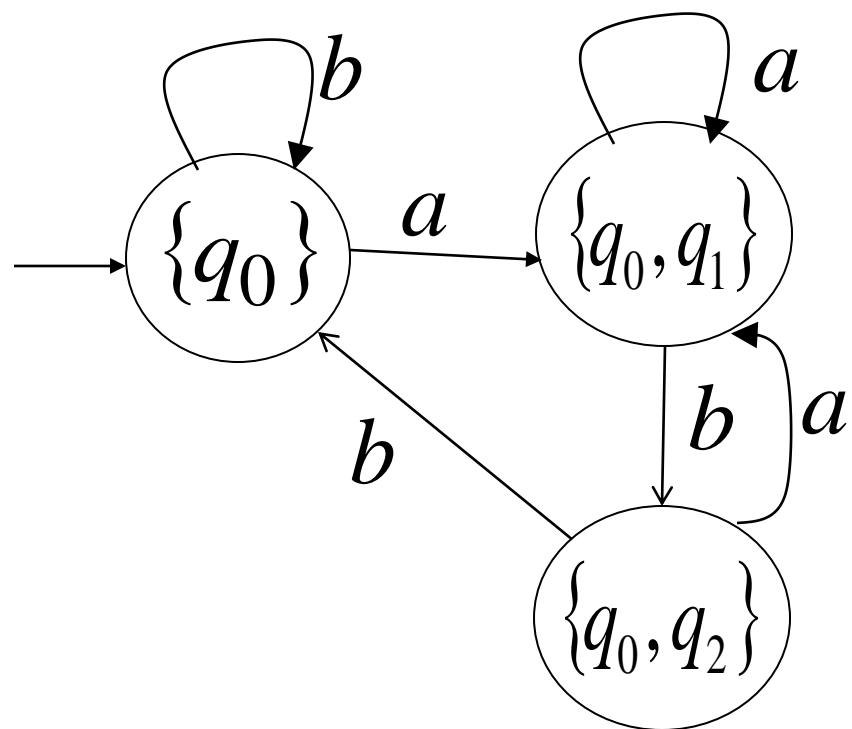


Figure 1

$$\delta^*(q_0, a) = \{q_0, q_1\}$$

$$\delta^*(\{q_0, q_1\}, a) = \delta^*(q_0, a) \cup \delta^*(q_1, a) = \{q_0, q_1\} \cup \Theta = \{q_0, q_1\}$$

$$\delta^*(\{q_0, q_1\}, b) = \delta^*(q_0, b) \cup \delta^*(q_1, b) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$$



$$\delta^*(\{q_0, q_2\}, a)$$

$$= \delta^*(q_0, a) \cup \delta^*(q_2, a)$$

$$= \{q_0, q_1\} \cup \{\Theta\} = \{q_0, q_1\}$$

$$\delta^*(\{q_0, q_2\}, b)$$

$$= \delta^*(q_0, b) \cup \delta^*(q_2, b)$$

$$= \{q_0\} \cup \{\Theta\} = \{q_0\}$$

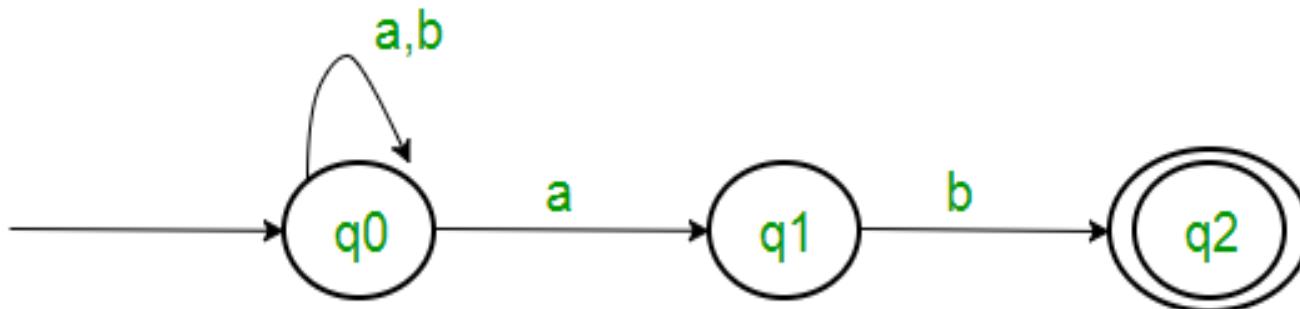
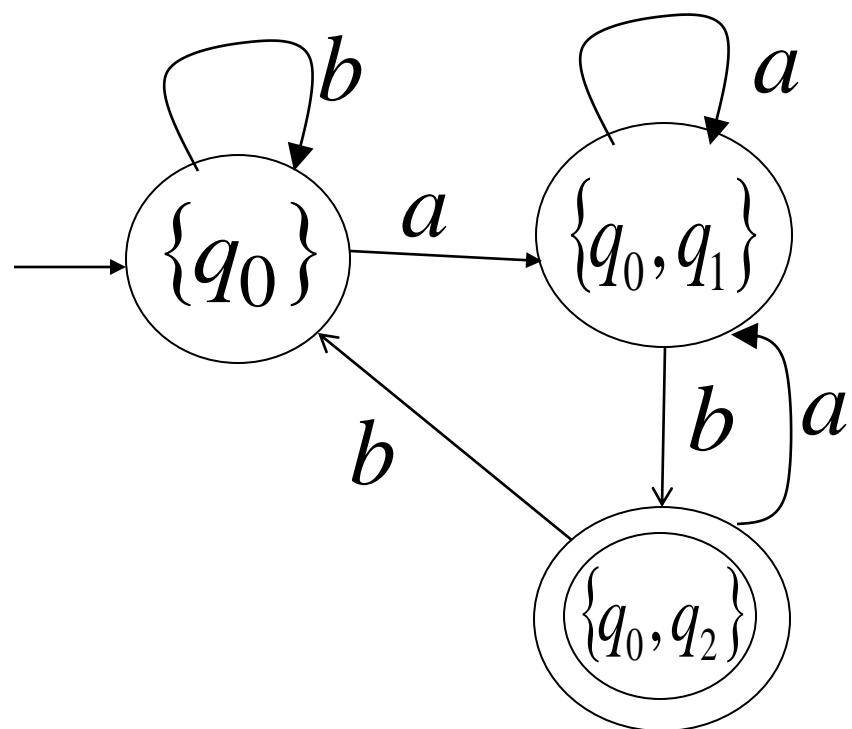


Figure 1



- Algorithm
- Input - An NDFA
- Output - An equivalent DFA
- Step 1 - Create state table from the given NDFA.
- Step 2 - Create a blank state table under possible input alphabets for the equivalent DFA.
- Step 3 - Mark the start state of the DFA by  $q_0$  (Same as the NDFA).
- Step 4 - Find out the combination of States  $\{Q_0, Q_1, \dots, Q_n\}$  for each possible input alphabet.
- Step 5 - Each time we generate a new DFA state under the input alphabet columns, we have to apply step 4 again, otherwise go to step 6.
- Step 6 - The states which contain any of the final states of the NDFA are the final states of the equivalent DFA.

# NFA

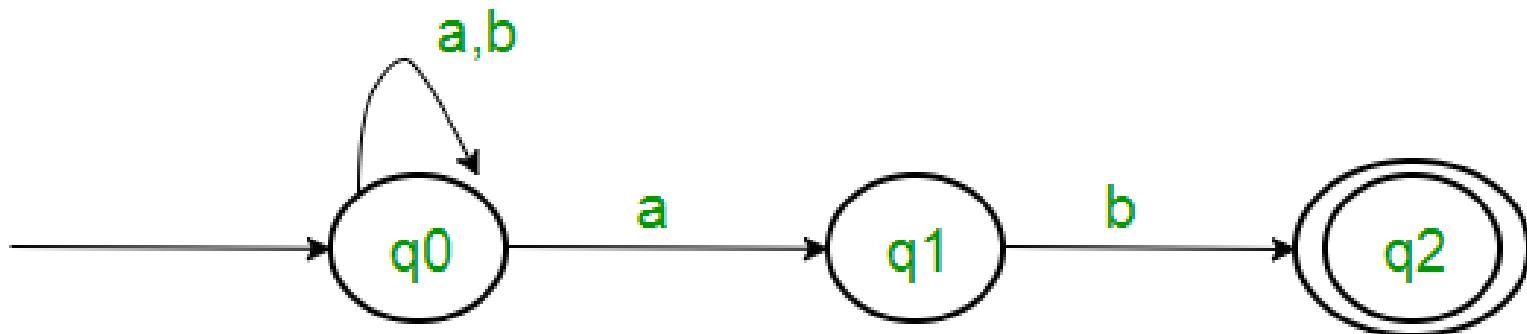


Figure 1

$$Q = \{q_0, q_1, q_2\}$$

$$\Sigma = a, b$$

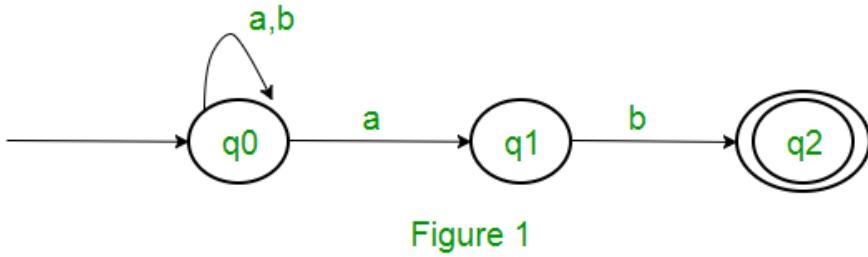
$$q_0 = q_0$$

$$F = \{q_2\}$$

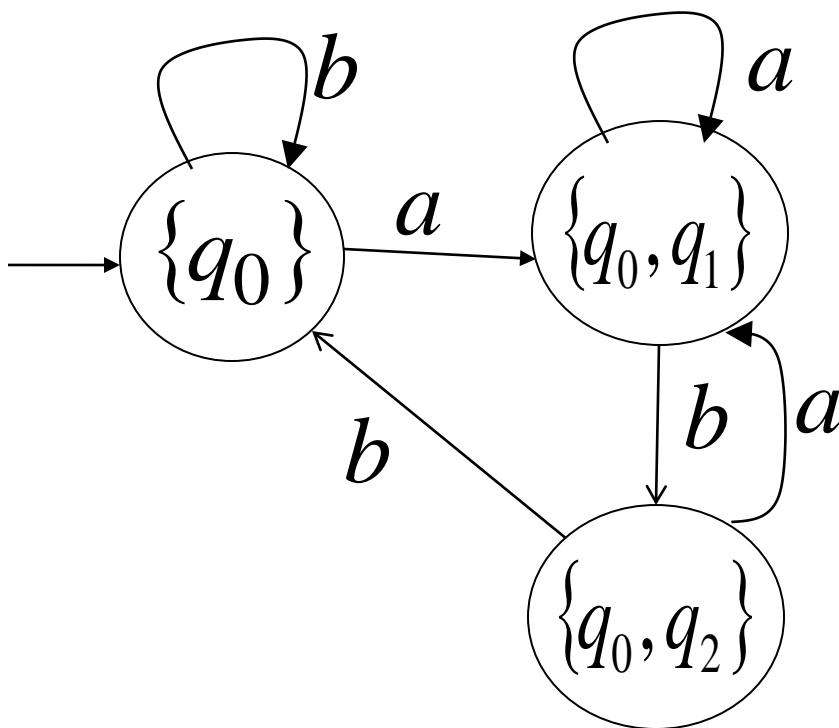
$\delta$  (Transition Function)

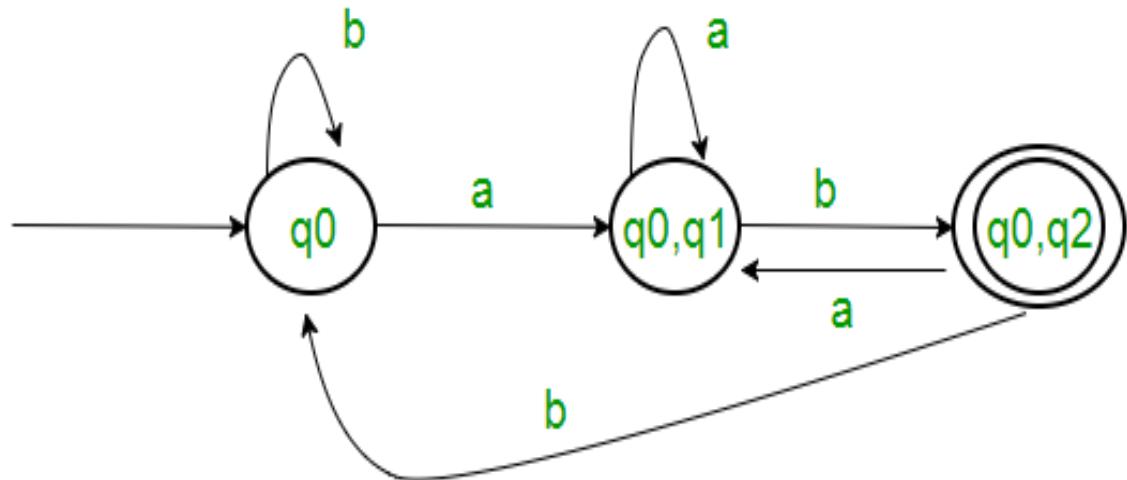
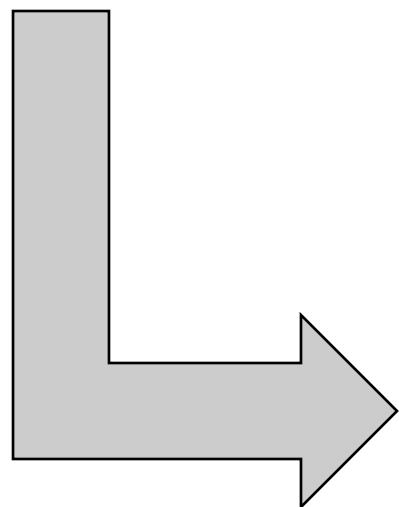
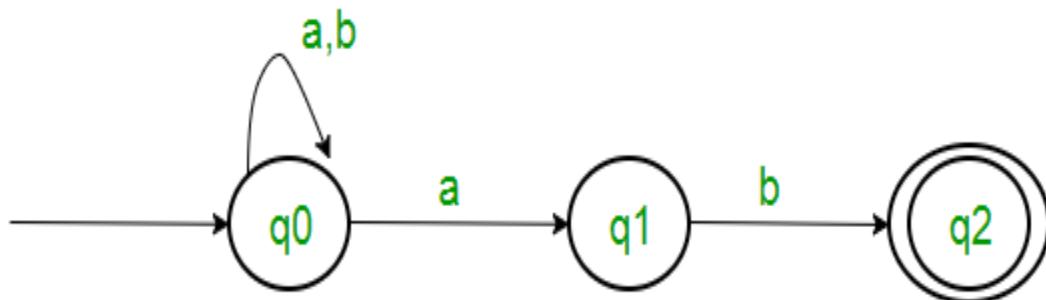
	$a$	$b$
$q_0$	$\{q_0, q_1\}$	$q_0$
$q_1$		$q_2$
$q_2$		

	a	b
q0	{q0,q1}	q0
q1		q2
q2		



State	a	B
q0	{q0,q1}	q0
{q0,q1}	{q0,q1}	{q0,q2}
{q0,q2}	{q0,q1}	q0





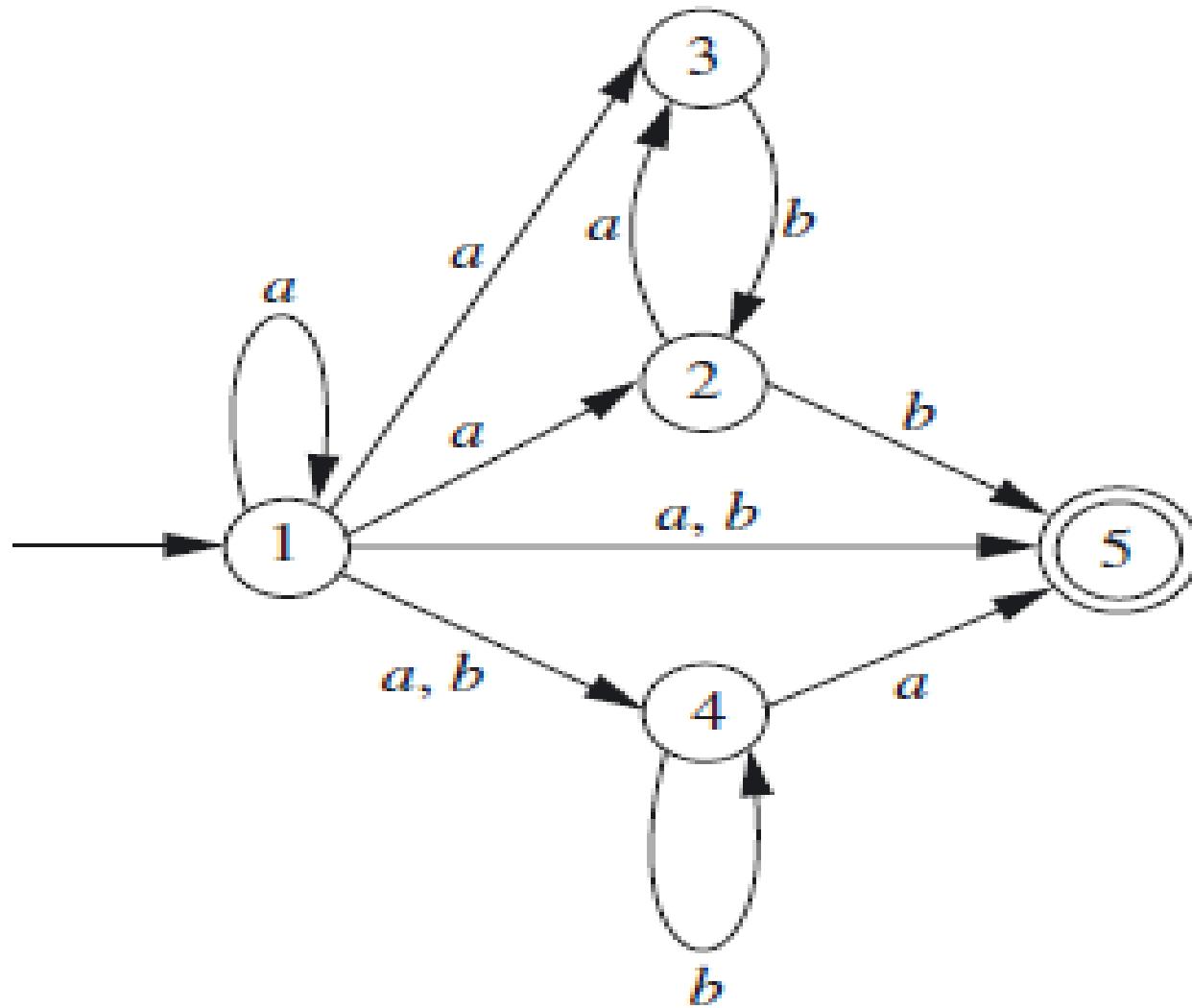
29/06/2019 Absent

- 5, 7, 9, 13, 16, 17, 21, 22, 26, 28, 32, 33, 34, 38, 39, 40, 41, 42, 43, 45, 46, 48, 51, 2, 4, 7, 9, 62, 5, 71, 72, 73, 74

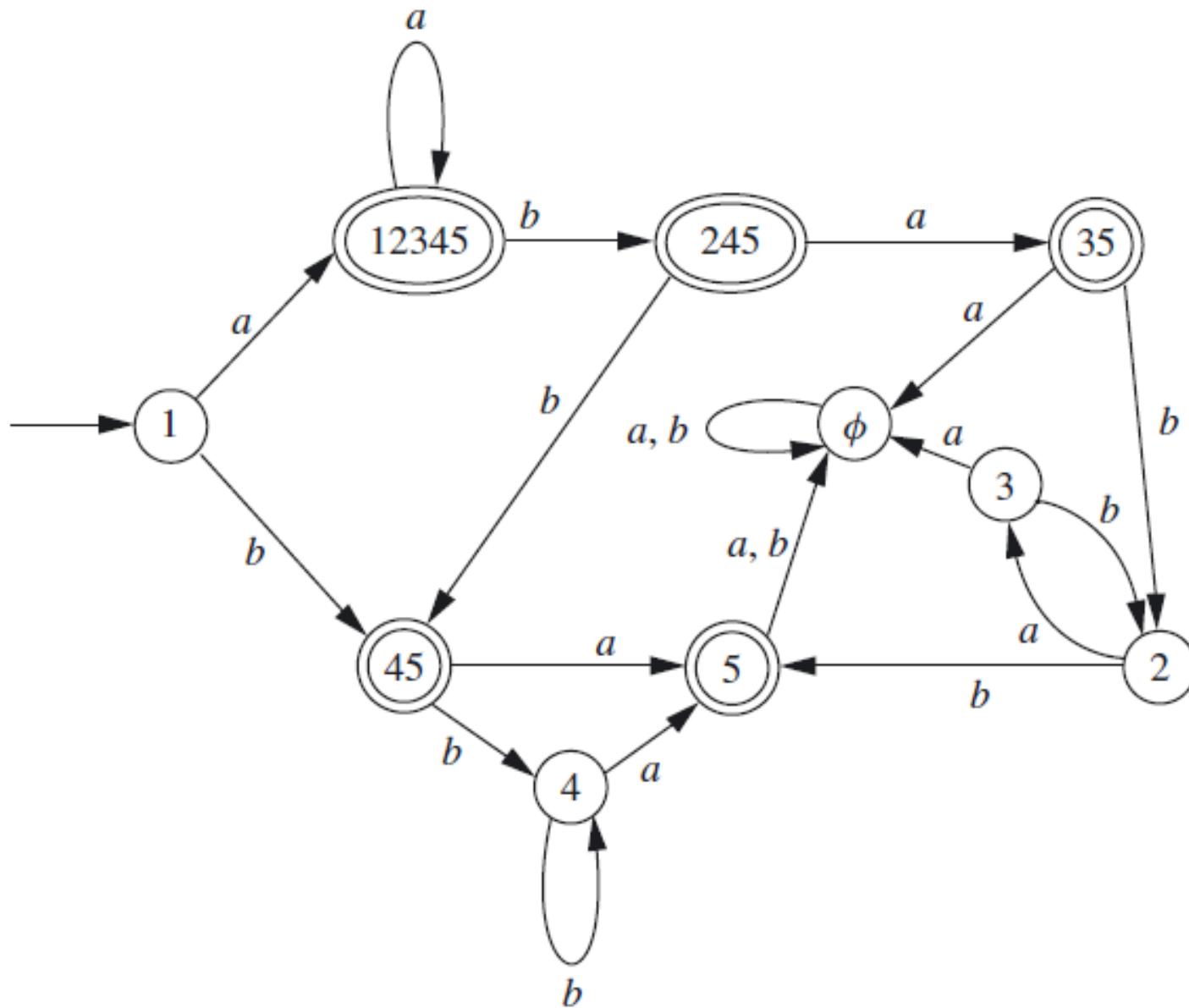
# RECALL

- Minimization of DFA
- DFA,NFA,NFA with Null accept regular language and Vice versa.
- DFA,NFA and NFA null Equivalent to each other.
- $\text{NFA} \rightarrow \text{DFA}$

# Example for NFA to DFA Conversion



# Output



# RECALL

- Minimization of DFA
- DFA,NFA,NFA with Null accept regular language and Vice versa.
- DFA,NFA and NFA null Equivalent to each other.
- $\text{NFA} \rightarrow \text{DFA}$
- $\text{NFA-null} \rightarrow \text{NFA}$
- $\text{NFA-null} \rightarrow \text{DFA}$

# THEORY OF COMPUTATION

---

Unit I

## FINITE STATE MACHINES

---

**Jagadish Kashinath Kamble** (M.Tech, IIT Kharagpur)

Dept. of Information Technology,  
Pune Institute of Computer Technology, Pune

# NFA- $\lambda$ to NFA Conversion

# Thomson Construction

- No Change In Initial State
- No Change In The Total No. Of States
- No change In Change In Final States

# General Conversion Procedure

**Input:** an NFA- $\lambda M$

**Output:** an equivalent NFA  $M'$   
with  $L(M) = L(M')$

Let  $M = (Q, \Sigma, \delta, q_0, F)$  -  $\epsilon$ -NFA

$M_1 = (Q_1, \Sigma, \delta_1, q_0', F_1)$  - NFA

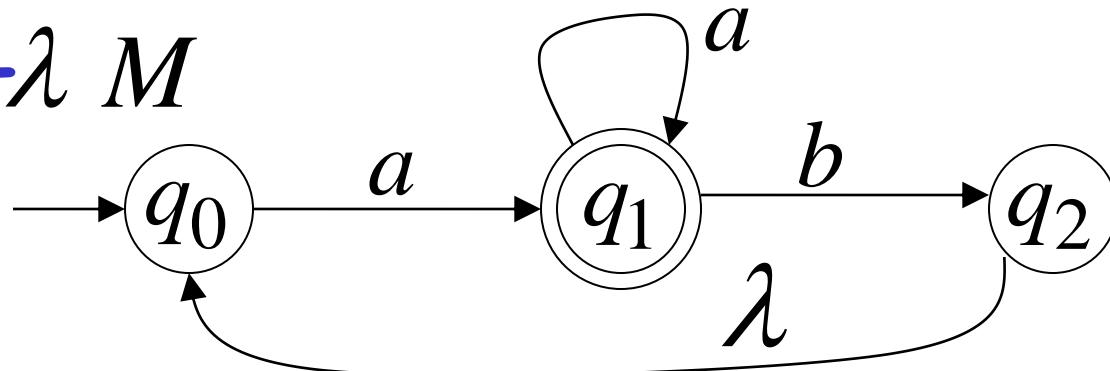
1) Initial State  $q_0' = q_0$

2) Construction Of  $\delta_1$

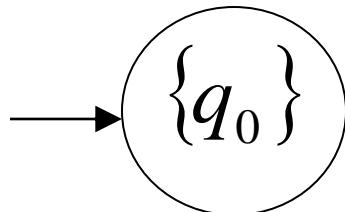
$$\delta_1(q, x) = \epsilon\text{-Closure}(\delta(\epsilon\text{-Closure}(q)), x)$$

# Conversion NFA- $\lambda$ to NFA

NFA- $\lambda$   $M$



NFA  $M'$



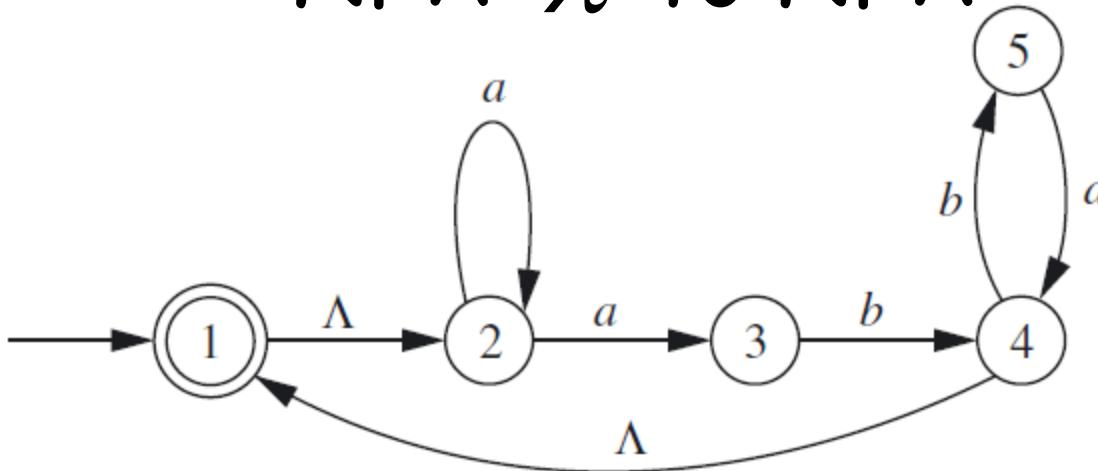
# The $\lambda$ -Closure of a Set of States

- Suppose  $M = (Q, \lambda, q_0, A, \delta)$  is an NFA, and  $S \subseteq Q$  is a set of states.
- The  $\lambda$ -closure of  $S$  is the set  $(S)$  that can be defined recursively as
  - follows.
    1.  $S \subseteq (S)$ .
    2. For every  $q \in (S)$ ,  $\delta(q, \lambda) \subseteq (S)$ .

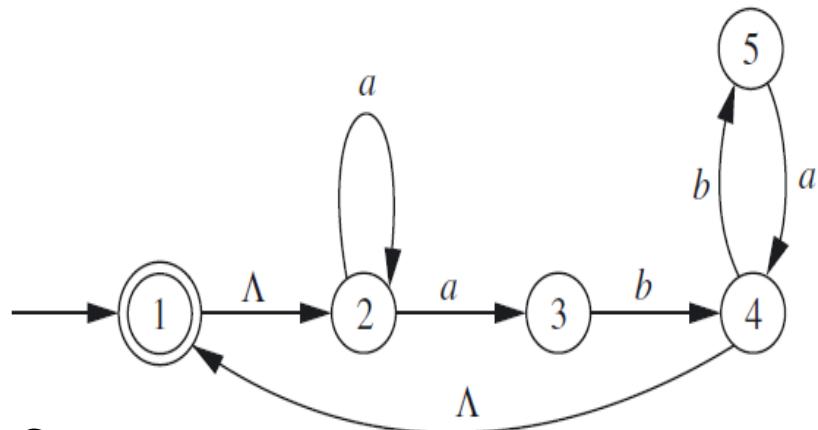
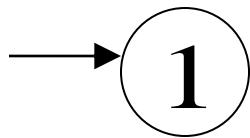
E.g.  $Q=\{1,2,3\}$ ,  $S=\{1\}$

null closure of  $S=(S)=\{1\}=\{1,2\}$

# NFA- $\lambda$ to NFA



1. States of NFA- $\lambda$  and NFA are Same
2. Initial state of NFA- $\lambda$  is initial state of NFA.
3. For transitions,
  - find the null closure of the state
  - apply the symbol from the alphabet
  - find the null closure of a set states generated from the above step
4. Repeat the step No.4 for each state



1. States of NFA-  $\lambda$  and NFA are Same
2. Initial state of NFA-  $\lambda$  is initial state of NFA.
3. For transitions,
  - find the null closure of the state
  - apply the symbol from the alphabet
  - find the null closure of the states generated from the above step

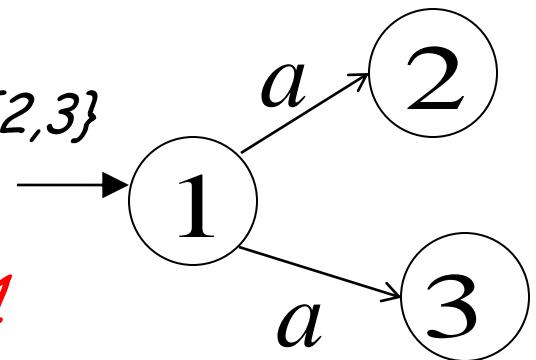
$$1) \lambda - (\{1\}) = \{1, 2\}$$

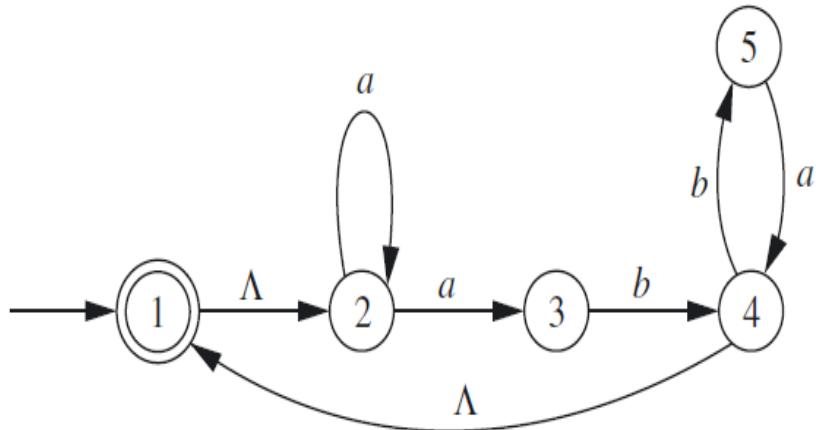
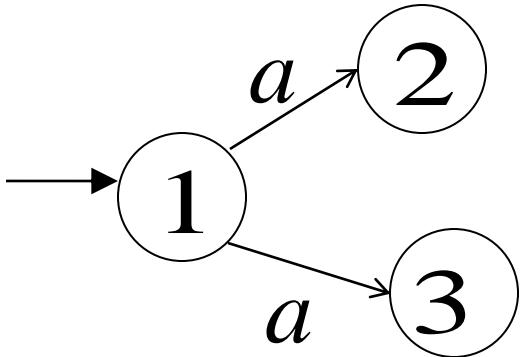
$$2) \delta(\{1, 2\}, a) = \delta(1, a) \cup \delta(2, a) = \emptyset \cup \{2, 3\} = \{2, 3\}$$

$$3) \lambda - \{2, 3\} = \{2, 3\}$$

4) After applying transitions  $a$  on state 1

It will move to state  $\{2, 3\}$





$$1) \lambda - \{1\} = \{1, 2\}$$

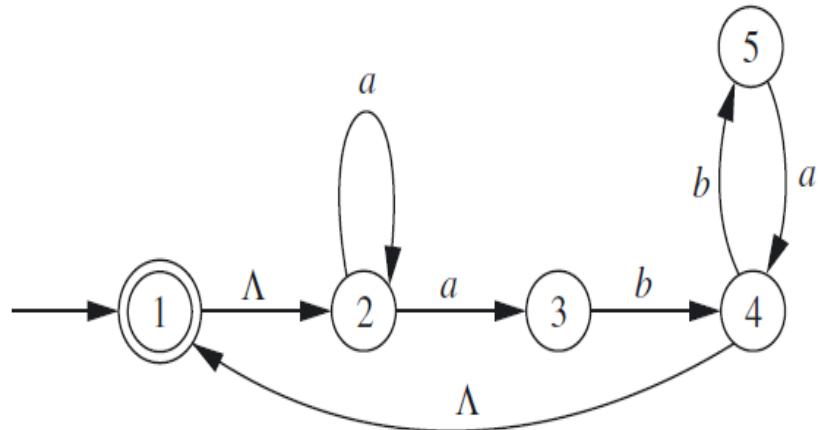
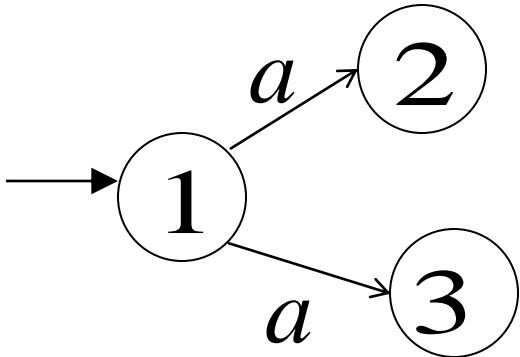
$$2) \delta(\{1, 2\}, a) = \delta(1, a) \cup \delta(2, a) = \emptyset \cup \{2, 3\}$$

$$3) \lambda - \{2, 3\} = \{2, 3\}$$

4) After applying transitions **a** on state **1**

It will move to state  $\{2, 3\}$

		<b>a</b>			<b>b</b>	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1, 2\}$	$\{2, 3\}$	$\{2, 3\}$			



$$1) \lambda - (\{1\}) = \{1, 2\}$$

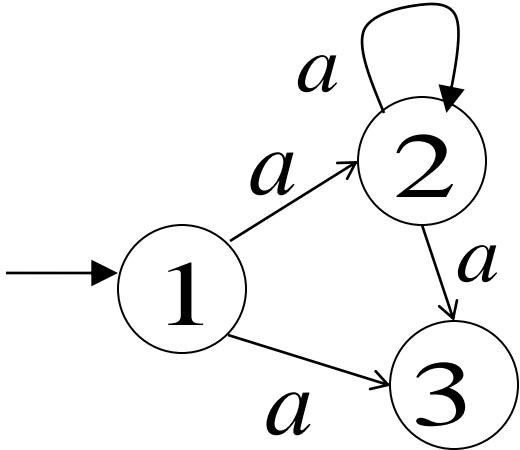
$$2) \delta(\{1, 2\}, b) = \delta(1, b) \cup \delta(2, b) = \emptyset \cup \emptyset$$

$$3) \lambda - \emptyset = \emptyset$$

4) After applying transitions  $b$  on state 1

It will move to the dead state

		a			b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2\}$	$\emptyset$	$\emptyset$



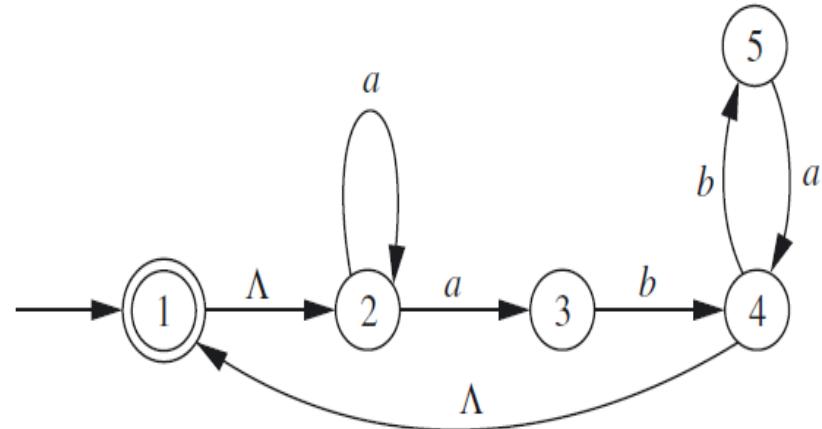
$$1) \lambda - (\{2\}) = \{2\}$$

$$2) \delta(\{2\}, a) = \{2, 3\}$$

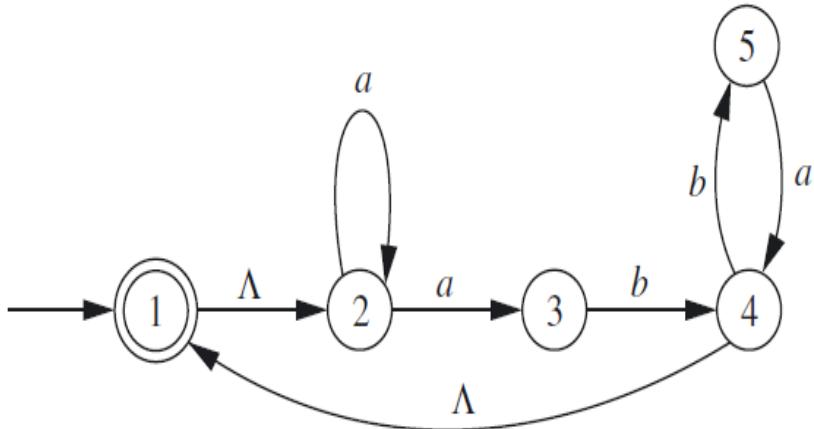
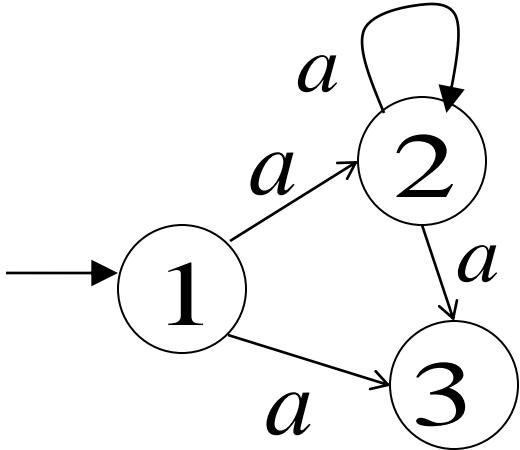
$$3) \lambda - \{2, 3\} = \{2, 3\}$$

4) After applying transitions  $a$  on state 2

It will move to state  $\{2, 3\}$



		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2\}$	$\emptyset$	$\emptyset$
2	$\{2\}$	$\{2, 3\}$	$\{2, 3\}$			



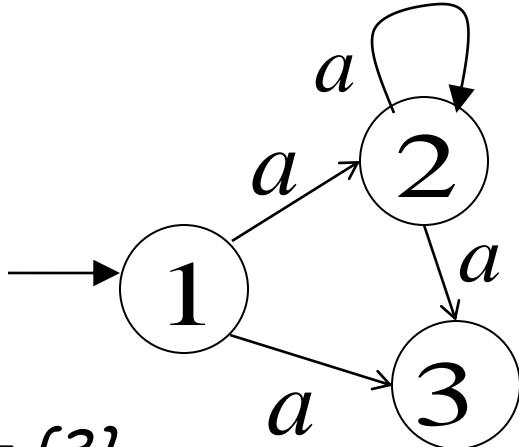
- 1)  $\lambda - (\{2\}) = \{2\}$
- 2)  $\delta(\{2\}, b) = \emptyset$

3)  $\lambda - \emptyset = \emptyset$

4) After applying transitions **b** on state **2**

It will move to Dead state.

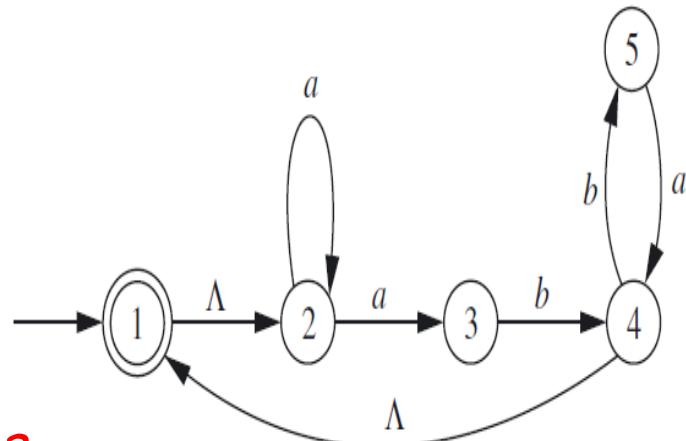
		a			b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2\}$	$\{2,3\}$	$\{2,3\}$	$\{1,2\}$	$\emptyset$	$\emptyset$
2	$\{2\}$	$\{2,3\}$	$\{2,3\}$	$\{2\}$	$\emptyset$	$\emptyset$



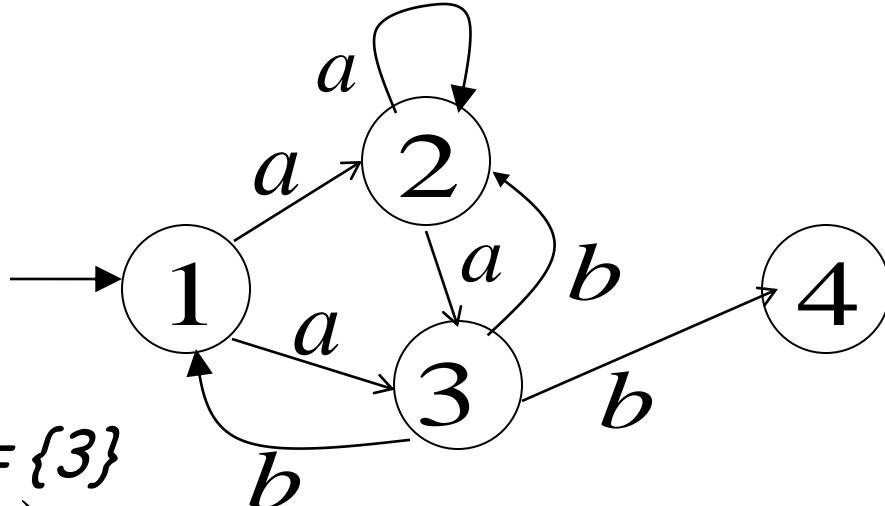
- 1)  $\lambda - (\{3\}) = \{3\}$
- 2)  $\delta(\{3\}, a) = \emptyset$
- 3)  $\lambda - \emptyset = \emptyset$

4) After applying transitions **a** on state 3

It will move to Dead State



		a			b	
1	$\{1,2\}$	$\{2,3\}$	$\{2,3\}$	$\{1,2\}$	$\emptyset$	$\lambda_{\emptyset}$
2	$\{2\}$	$\{2,3\}$	$\{2,3\}$	$\{2\}$	$\emptyset$	$\emptyset$
3	$\{3\}$	$\emptyset$	$\emptyset$			
4						
5						



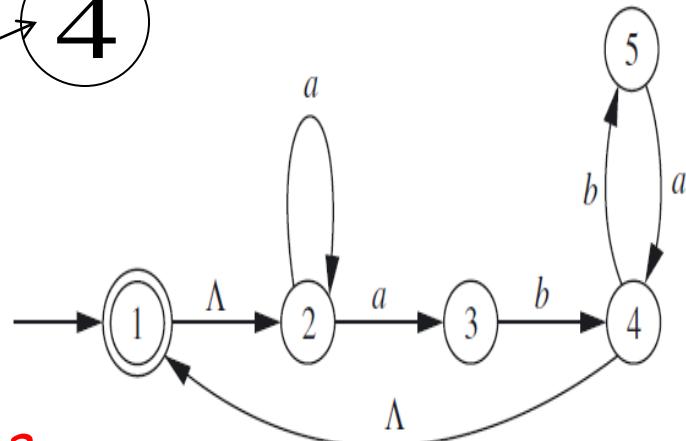
$$1) \lambda - \{3\} = \{3\}$$

$$2) \delta(\{3\}, b) = 4$$

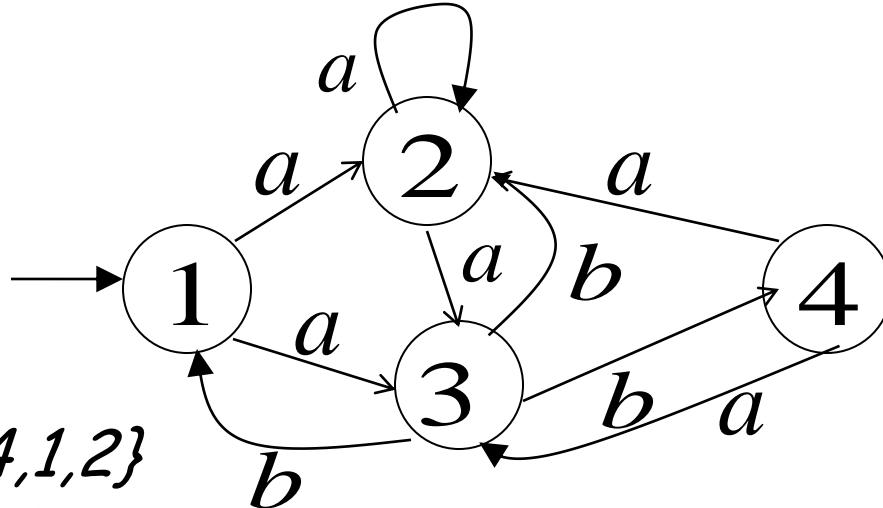
$$3) \lambda - \{4\} = \{4, 1, 2\}$$

4) After applying transitions  $b$  on state 3

It will move to State 1, 2 and 4



	$a$				$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2\}$	$\emptyset$	$\emptyset$
2	$\{2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{2\}$	$\emptyset$	$\emptyset$
3	$\{3\}$	$\emptyset$	$\emptyset$	$\{3\}$	$\{4\}$	$\{4, 1, 2\}$
4						
5						



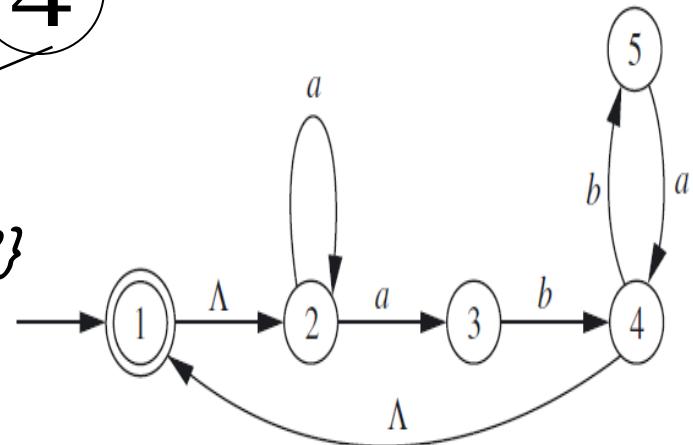
$$1) \lambda - \{4\} = \{4, 1, 2\}$$

$$2) \delta(\{4, 1, 2\}, a) = \delta(4, a) \cup \delta(1, a) \cup \delta(2, a) = \{2, 3\}$$

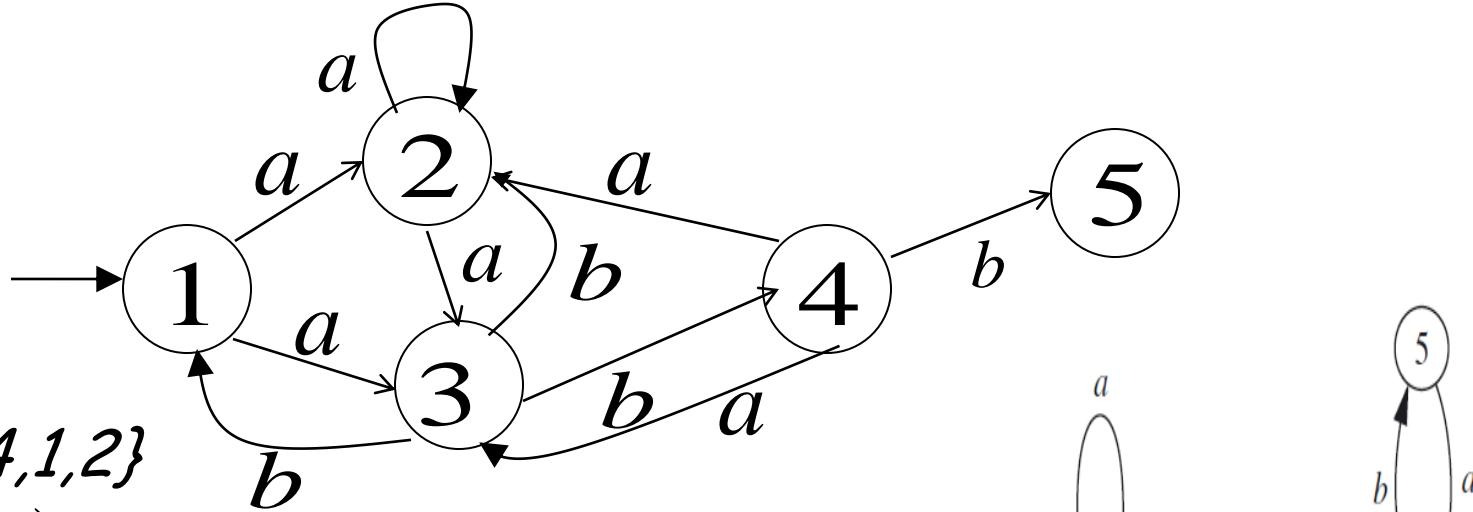
$$3) \lambda - \{2, 3\} = \{2, 3\}$$

4) After applying transitions **a** on state **4**

It will move to State **2 and 3**



	<b>a</b>				<b>b</b>	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2\}$	$\emptyset$	$\emptyset$
2	$\{2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{2\}$	$\emptyset$	$\emptyset$
3	$\{3\}$	$\emptyset$	$\emptyset$	$\{3\}$	$\{4\}$	$\{4, 1, 2\}$
4	$\{4, 1, 2\}$	$\{2, 3\}$	$\{2, 3\}$			
5						



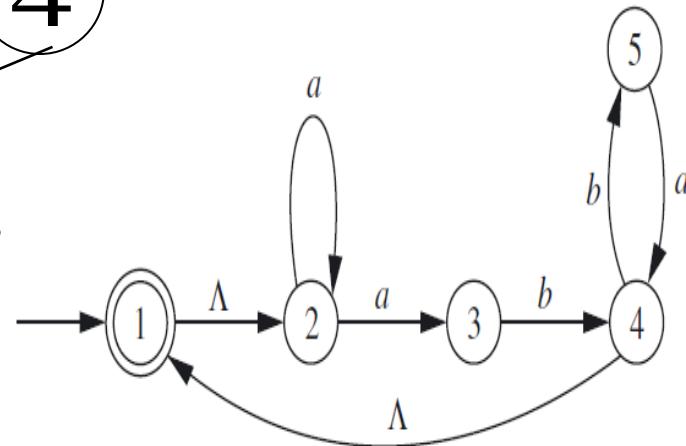
$$1) \lambda - \{4\} = \{4, 1, 2\}$$

$$2) \delta(\{4, 1, 2\}, b) = \delta(4, b) \cup \delta(1, b) \cup \delta(2, b) = \{5\}$$

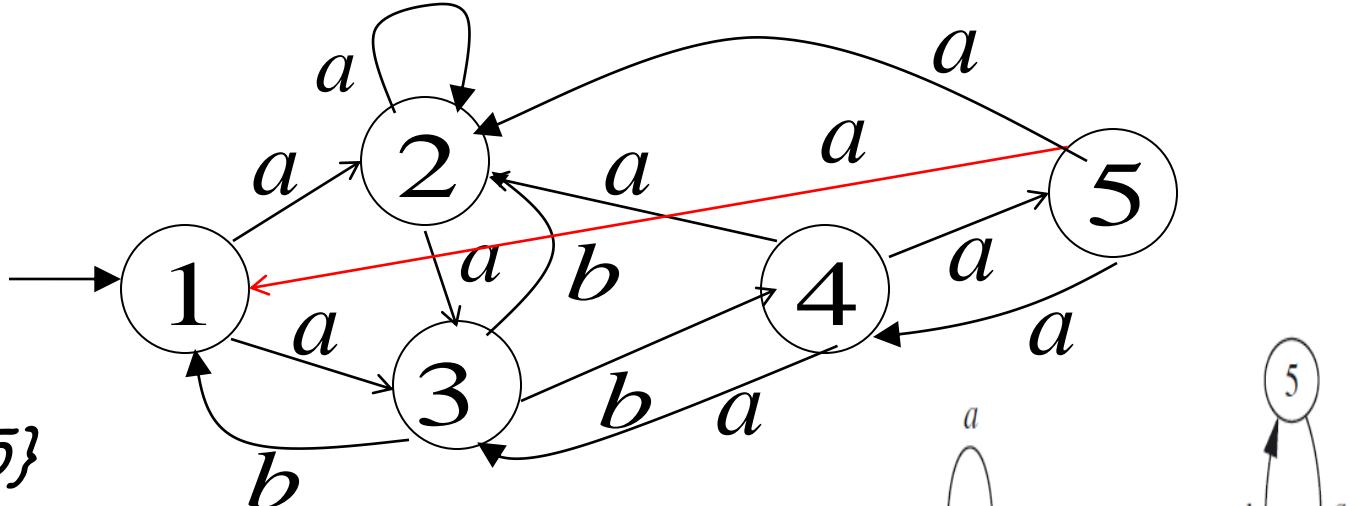
$$3) \lambda - \{5\} = \{5\}$$

4) After applying transitions  $b$  on state 4

It will move to State 5



	$a$				$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2\}$	$\emptyset$	$\emptyset$
2	$\{2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{2\}$	$\emptyset$	$\emptyset$
3	$\{3\}$	$\emptyset$	$\emptyset$	$\{3\}$	$\{4\}$	$\{4, 1, 2\}$
4	$\{4, 1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2, 4\}$	$\{5\}$	$\{5\}$
5						



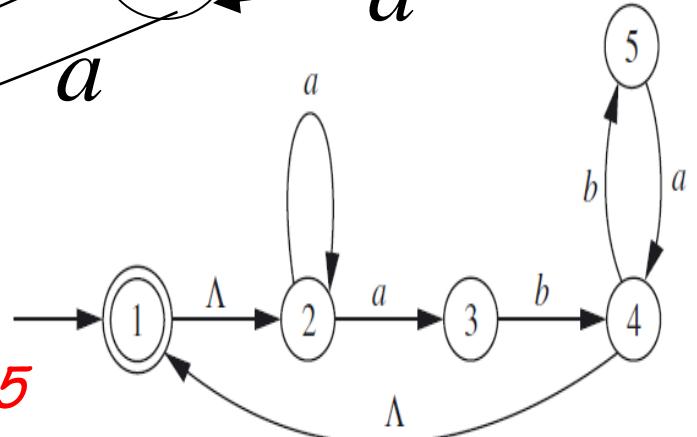
$$1) \lambda - \{5\} = \{5\}$$

$$2) \delta(\{5\}, a) = \{4\}$$

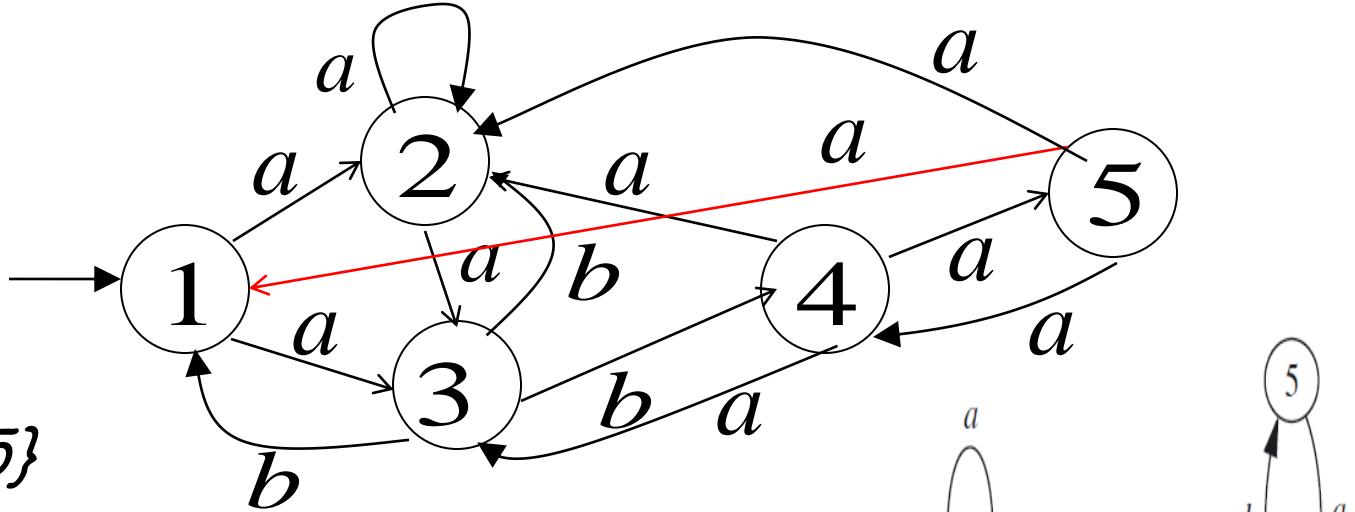
$$3) \lambda - \{4\} = \{1, 2, 4\}$$

4) After applying transitions  $a$  on state  $5$

It will move to State  $1, 2$  and  $4$



	$a$			$b$		
	$\lambda$		$\lambda$	$\lambda$		
1	$\{1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2\}$	$\emptyset$	$\emptyset$
2	$\{2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{2\}$	$\emptyset$	$\emptyset$
3	$\{3\}$	$\emptyset$	$\emptyset$	$\{3\}$	$\{4\}$	$\{4, 1, 2\}$
4	$\{4, 1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2, 4\}$	$\{5\}$	$\{5\}$
5	$\{5\}$	$\{4\}$	$\{1, 2, 4\}$			



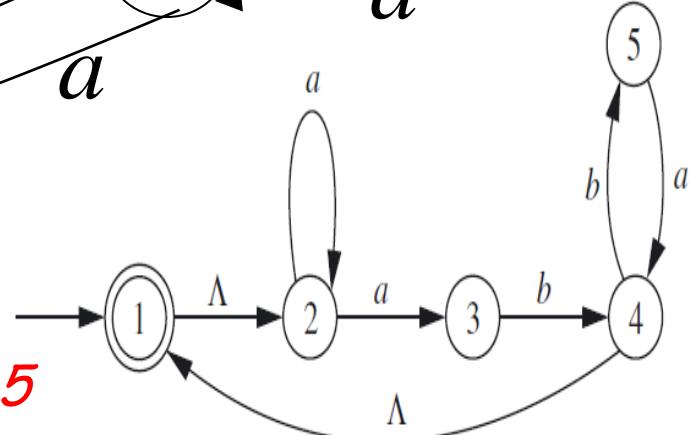
$$1) \lambda - \{5\} = \{5\}$$

$$2) \delta(\{5\}, b) = \{\emptyset\}$$

$$3) \lambda - \{\emptyset\} = \{\emptyset\}$$

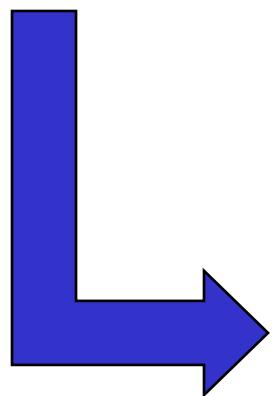
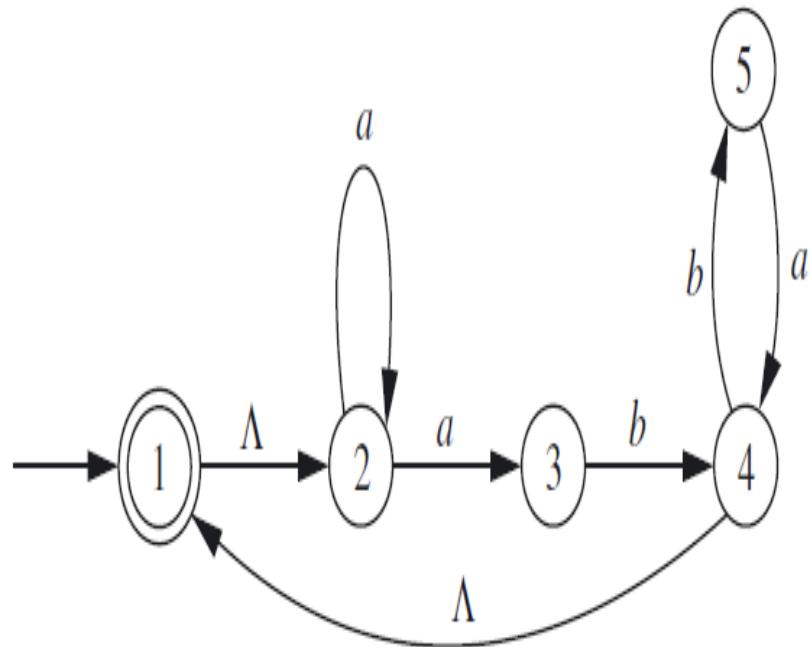
4) After applying transitions  $b$  on state 5

It will move to Dead State

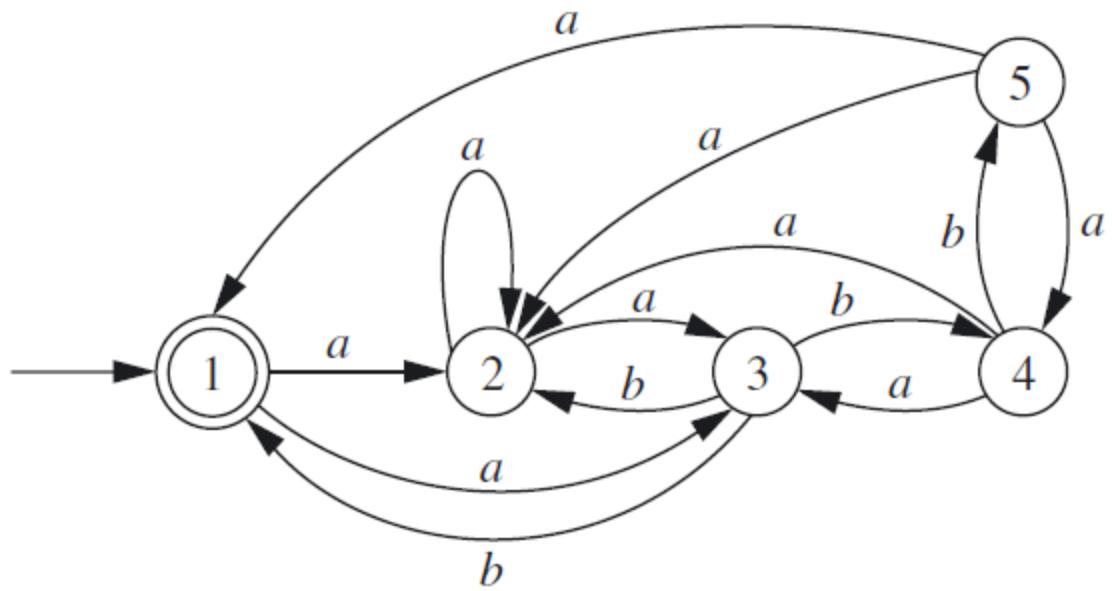


	a				b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2\}$	$\{2,3\}$	$\{2,3\}$	$\{1,2\}$	$\emptyset$	$\emptyset$
2	$\{2\}$	$\{2,3\}$	$\{2,3\}$	$\{2\}$	$\emptyset$	$\emptyset$
3	$\{3\}$	$\emptyset$	$\emptyset$	$\{3\}$	$\{4\}$	$\{4,1,2\}$
4	$\{4,1,2\}$	$\{2,3\}$	$\{2,3\}$	$\{1,2,4\}$	$\{5\}$	$\{5\}$
5	$\{5\}$	$\{4\}$	$\{1,2,4\}$	$\{5\}$	$\emptyset$	$\emptyset$

NFA- $\lambda$

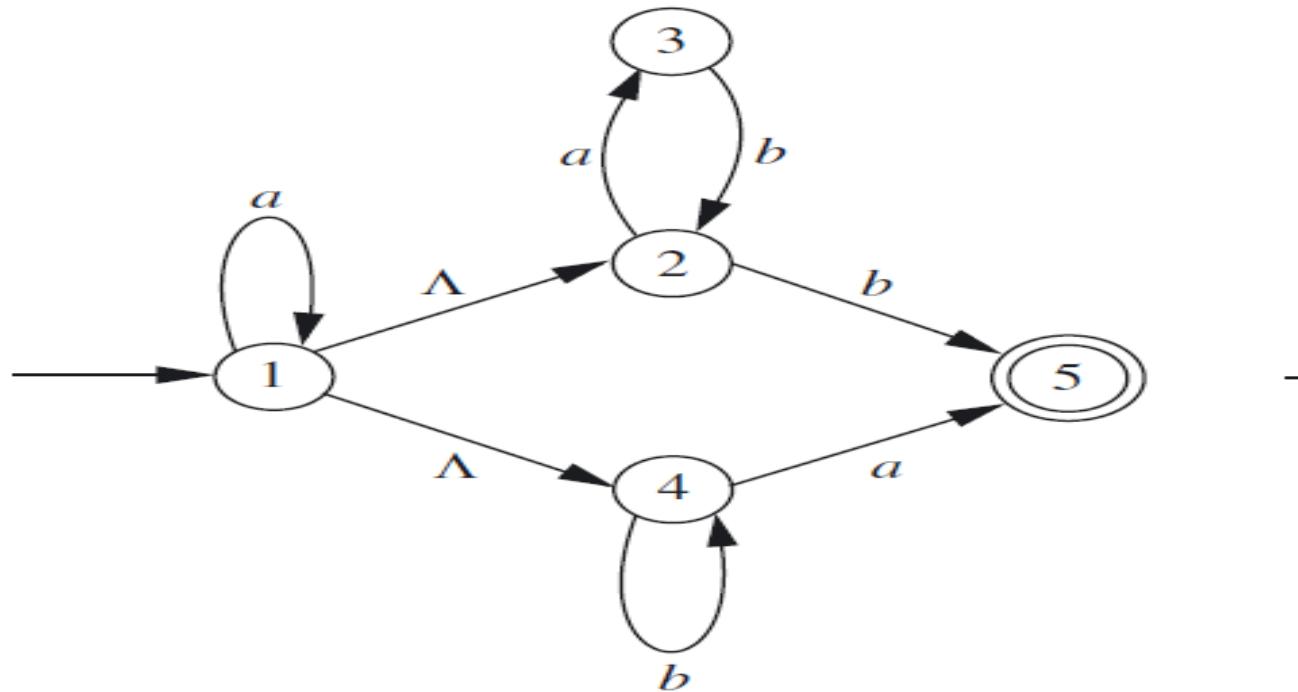


NFA

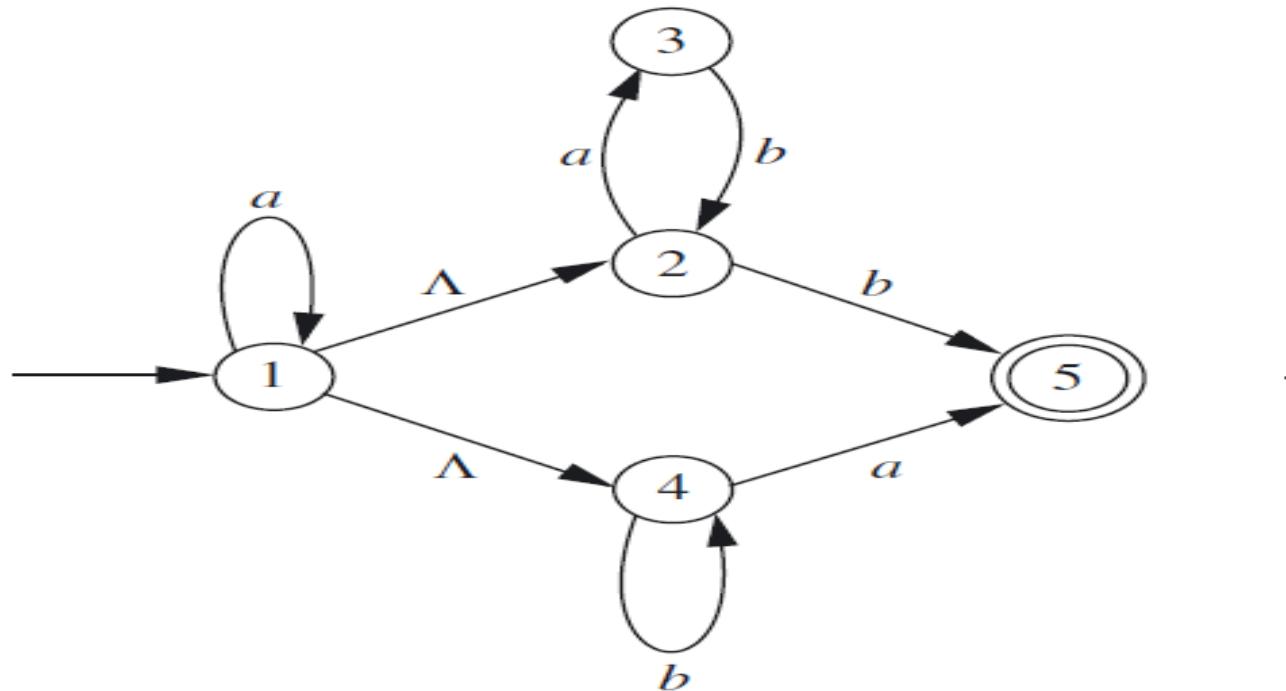


# RECALL

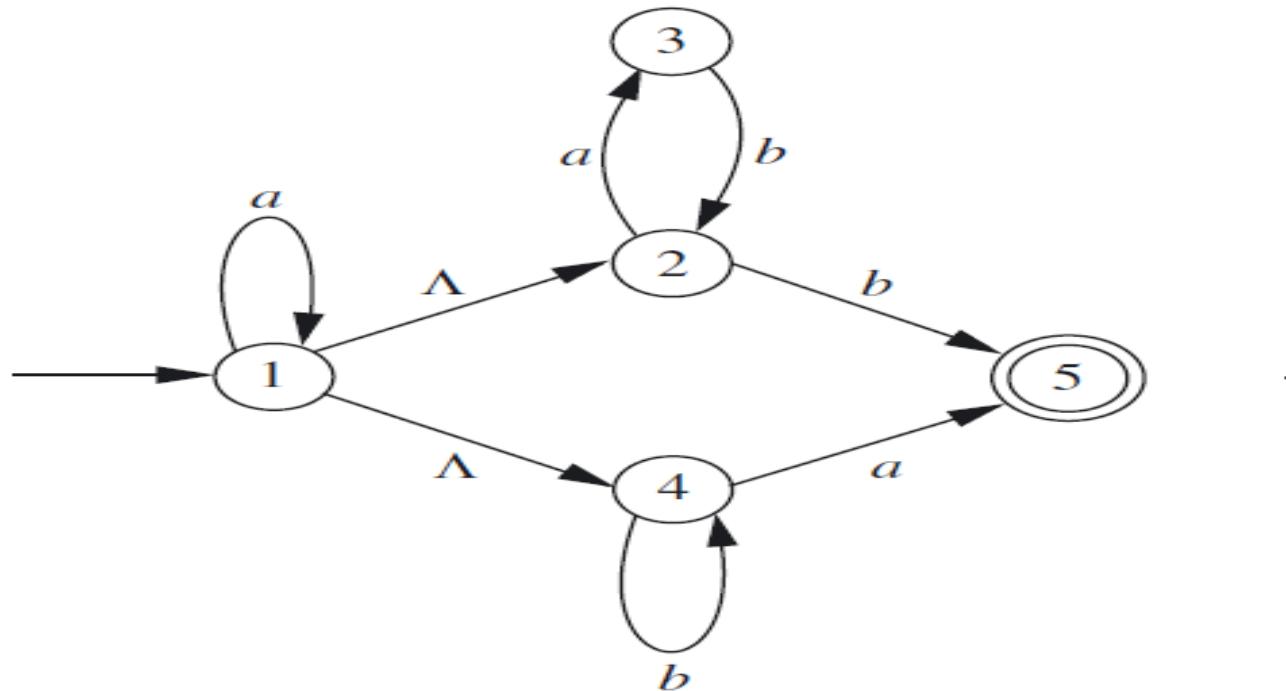
- Minimization of DFA
- DFA,NFA,NFA with Null accept regular language and Vice versa.
- DFA,NFA and NFA null Equivalent to each other.
- $\text{NFA} \rightarrow \text{DFA}$
- $\text{NFA-null} \rightarrow \text{NFA}$
- $\text{NFA-null} \rightarrow \text{DFA}$



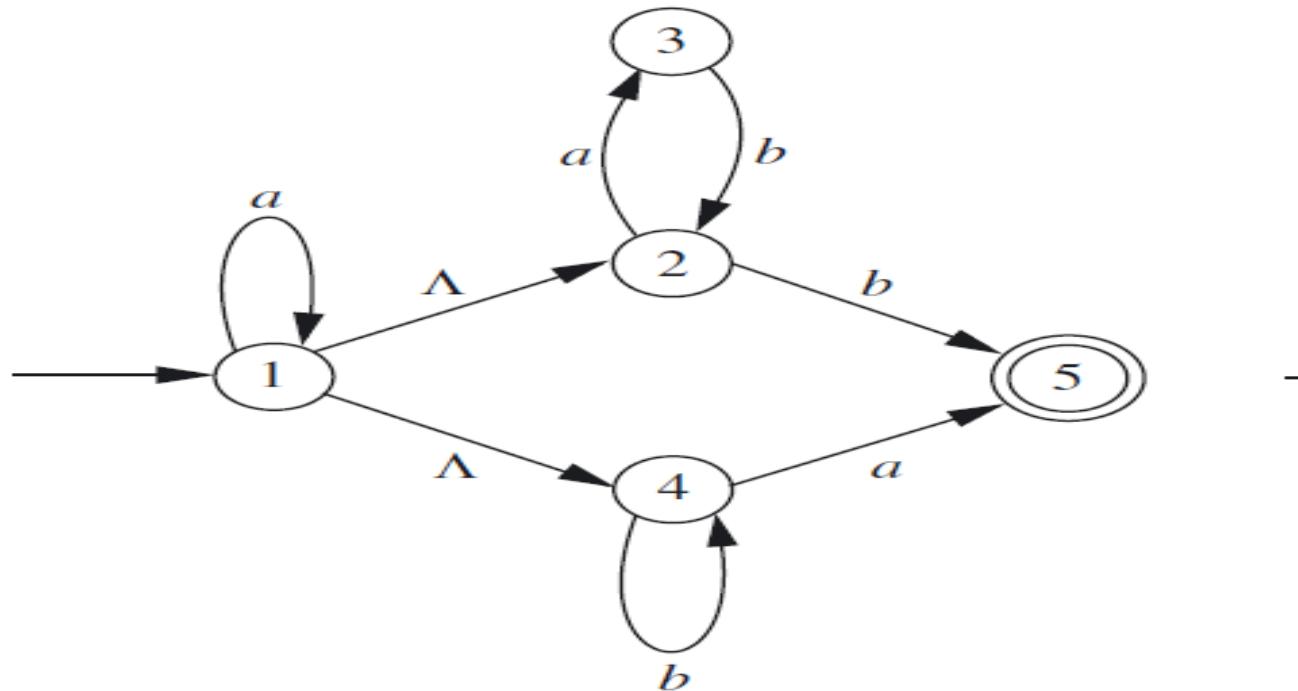
	$\lambda$	$a$			$b$	
1	$\lambda$		$\lambda$	$\lambda$		$\lambda$
2						
3						
4						
5						



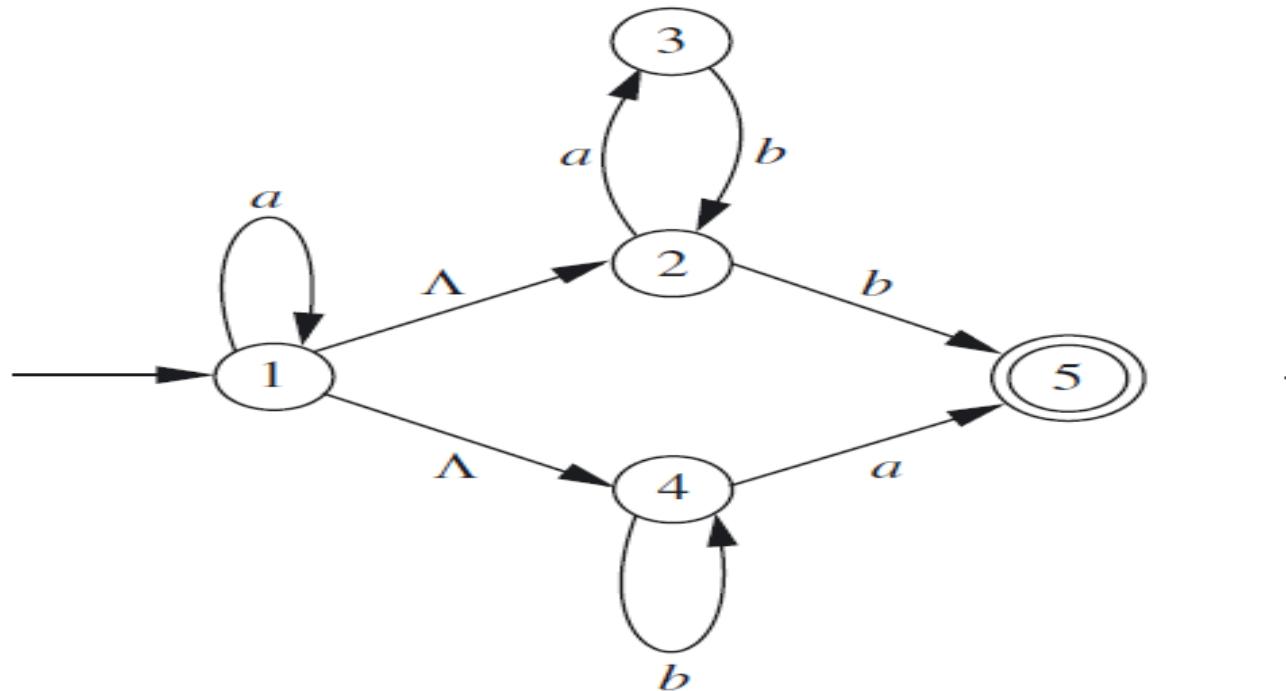
		a			b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,					
2						
3						
4						
5						



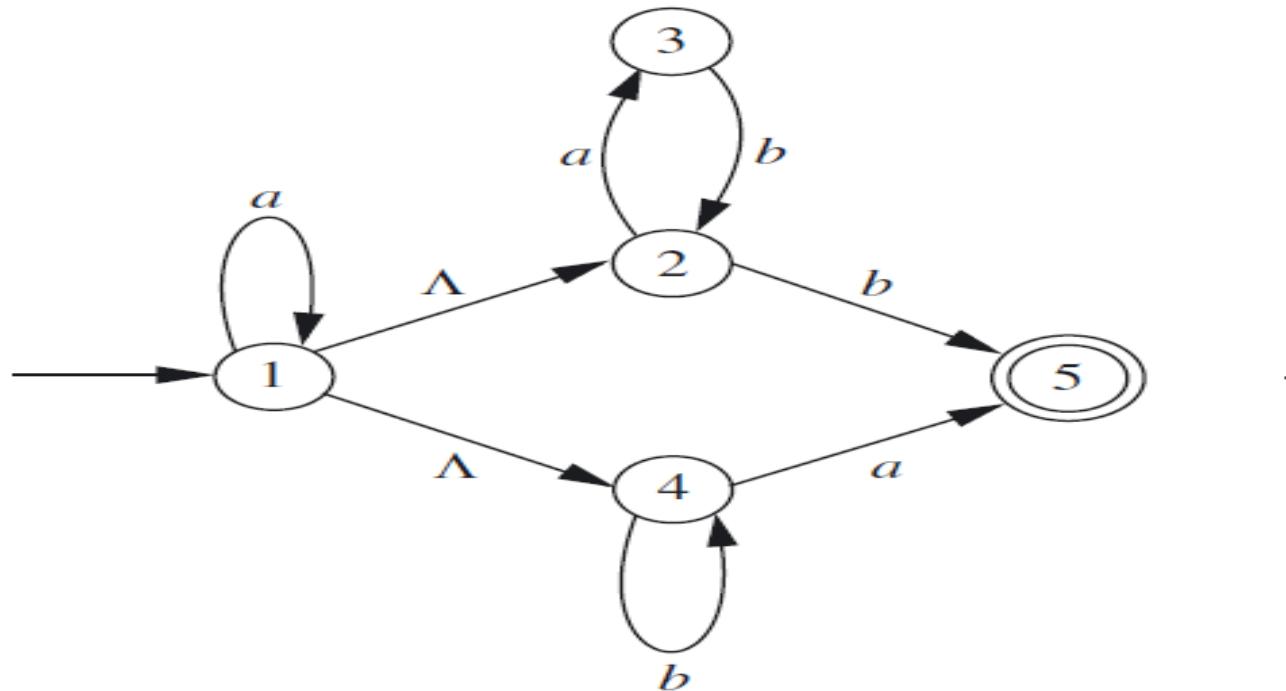
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\lambda$	$\lambda$	$\lambda$	$\lambda$
2						
3						
4						
5						



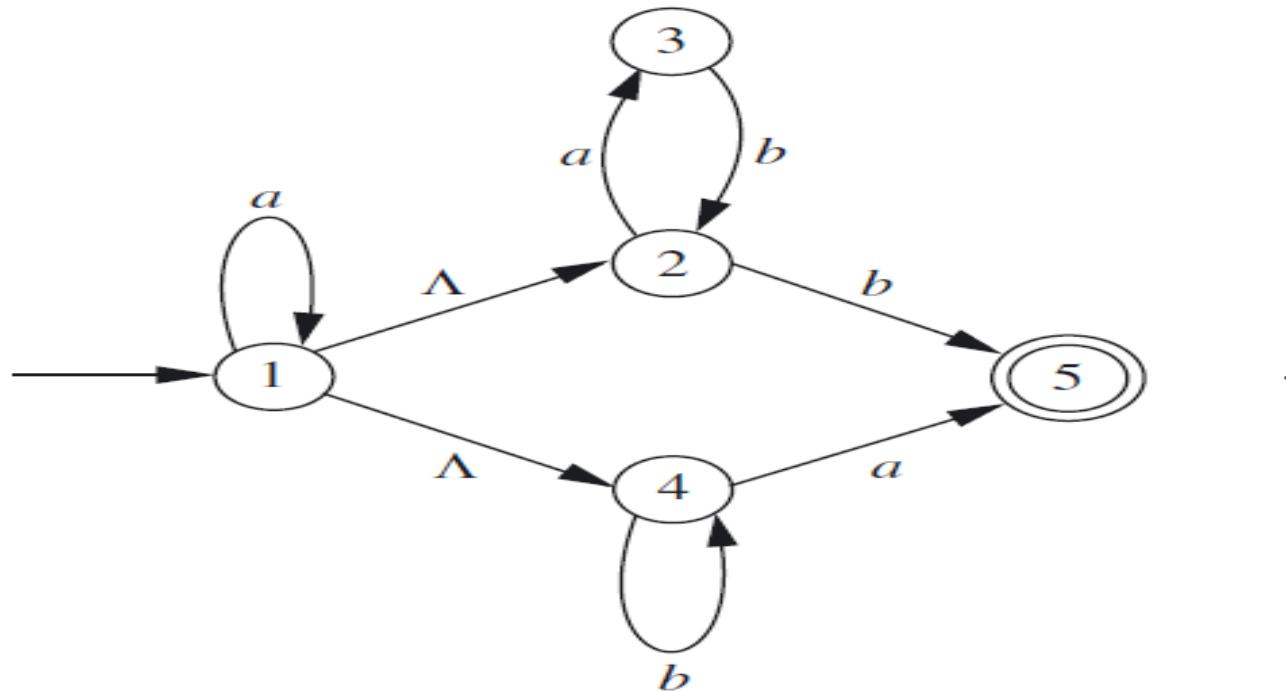
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$			$\lambda$
2						
3						
4						
5						



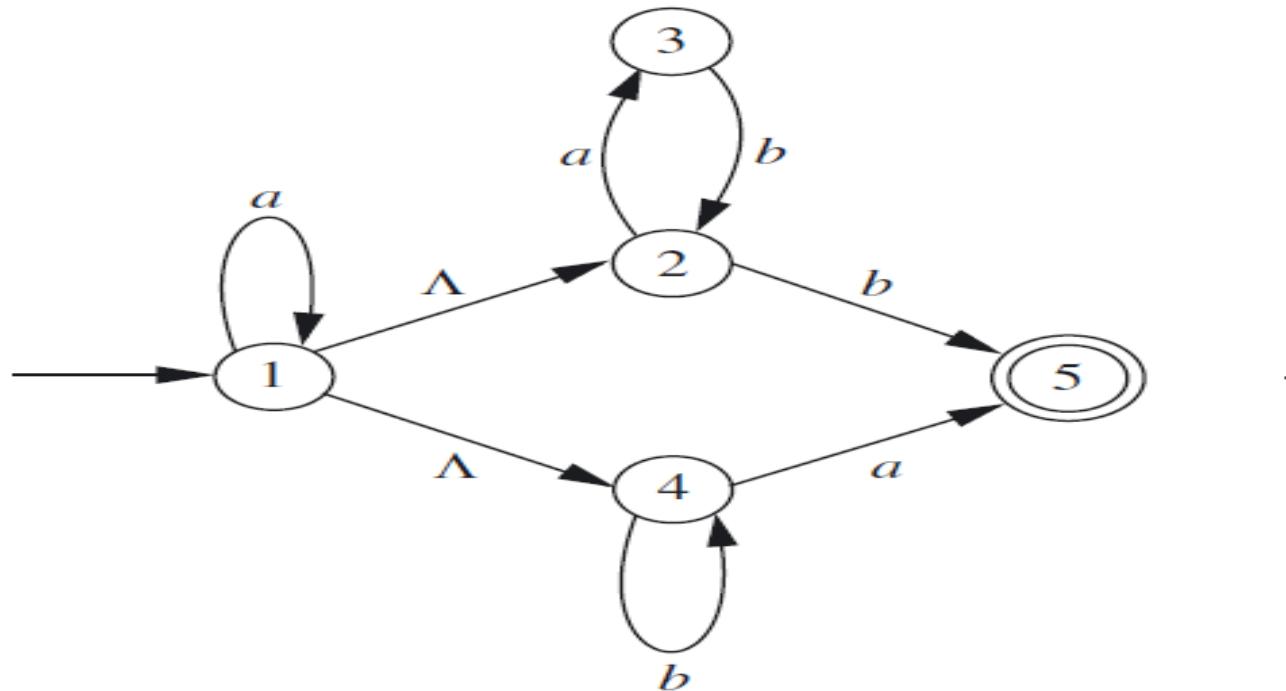
	$\lambda$	$a$			$b$	
	$\lambda$	$\lambda$	$\lambda$	$\lambda$	$\lambda$	$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$		
2						
3						
4						
5						



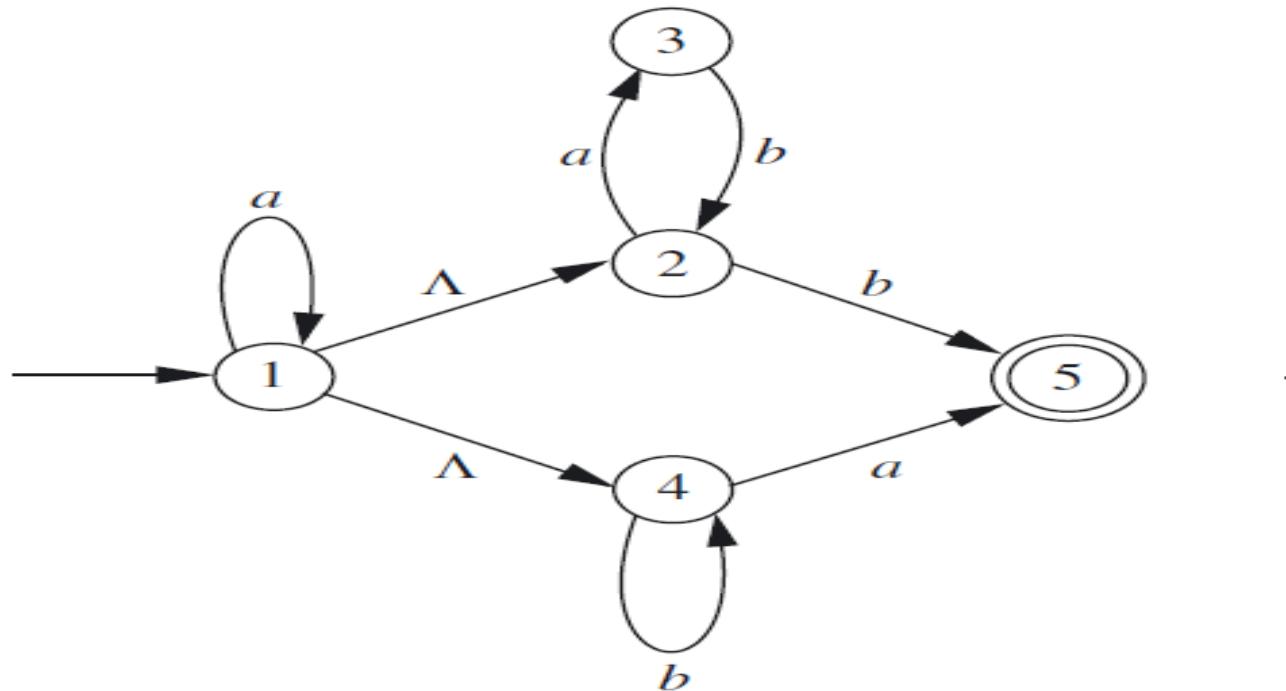
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	
2						
3						
4						
5						



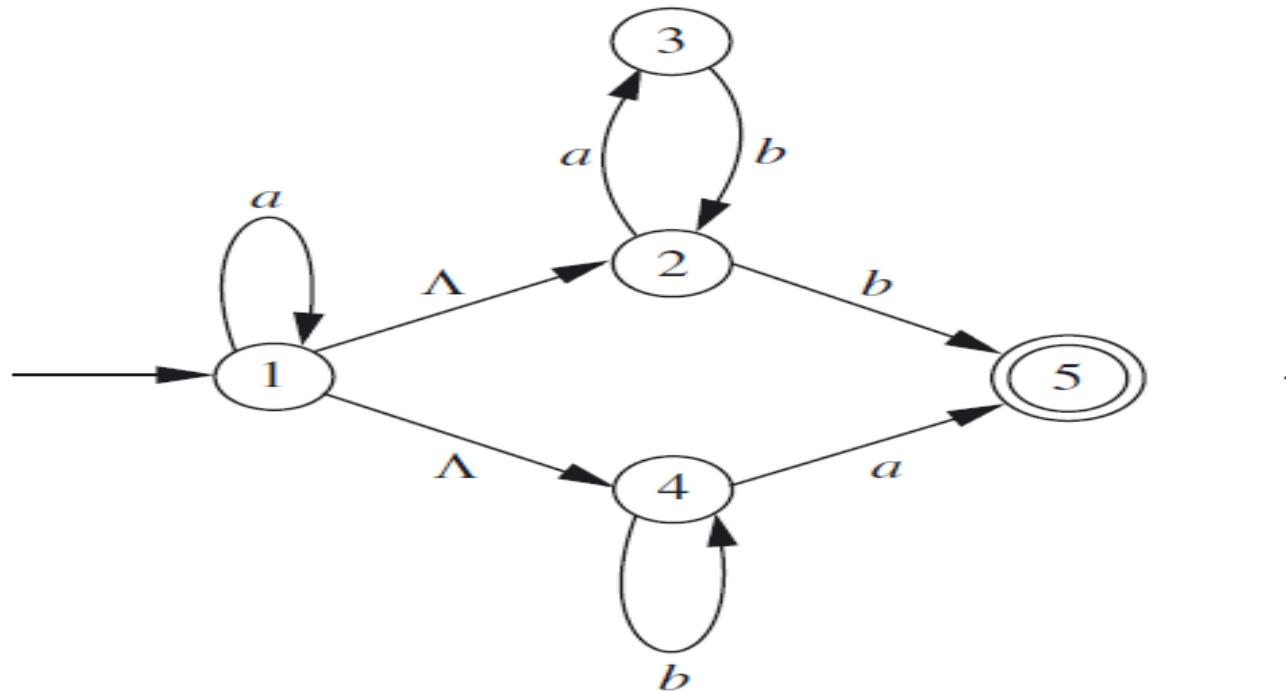
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2						
3						
4						
5						



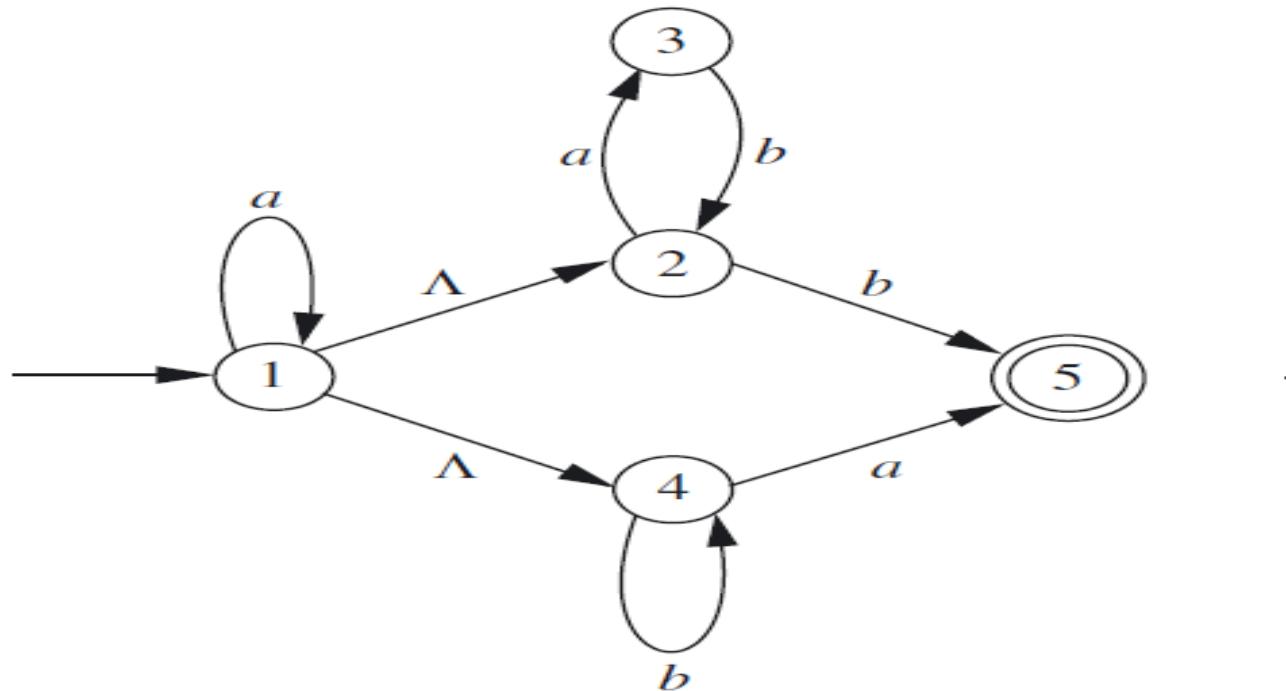
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$					
3						
4						
5						



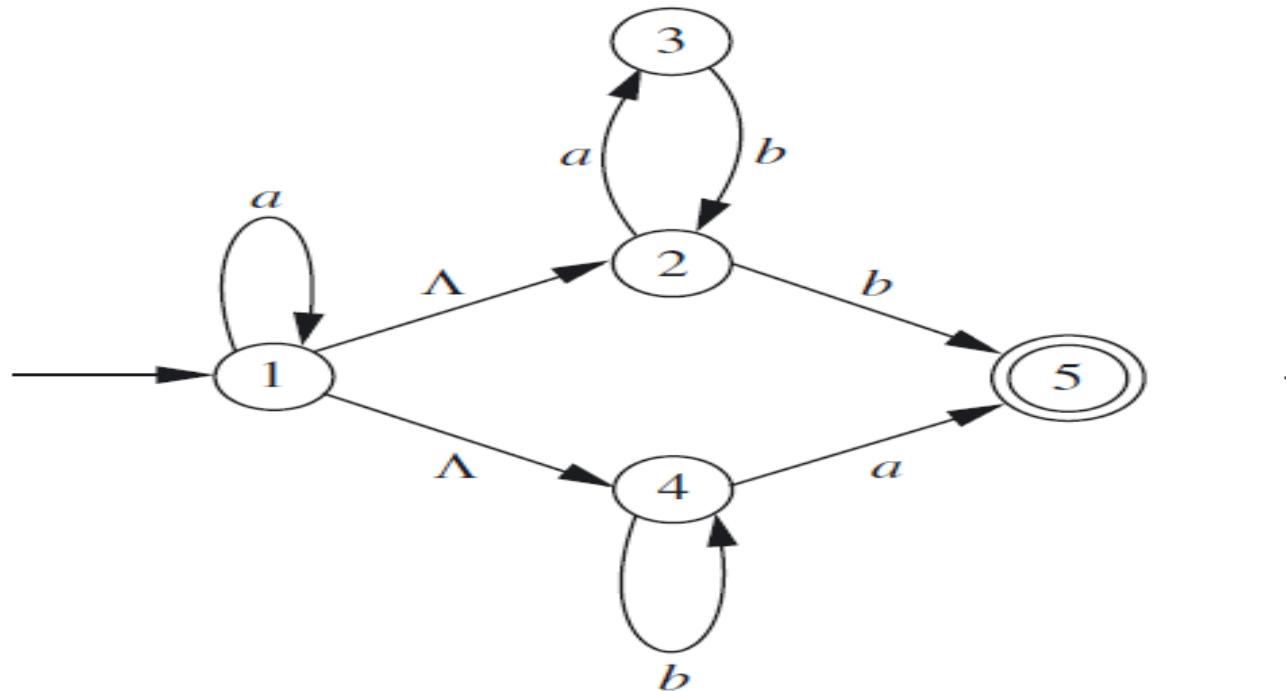
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$				
3						
4						
5						



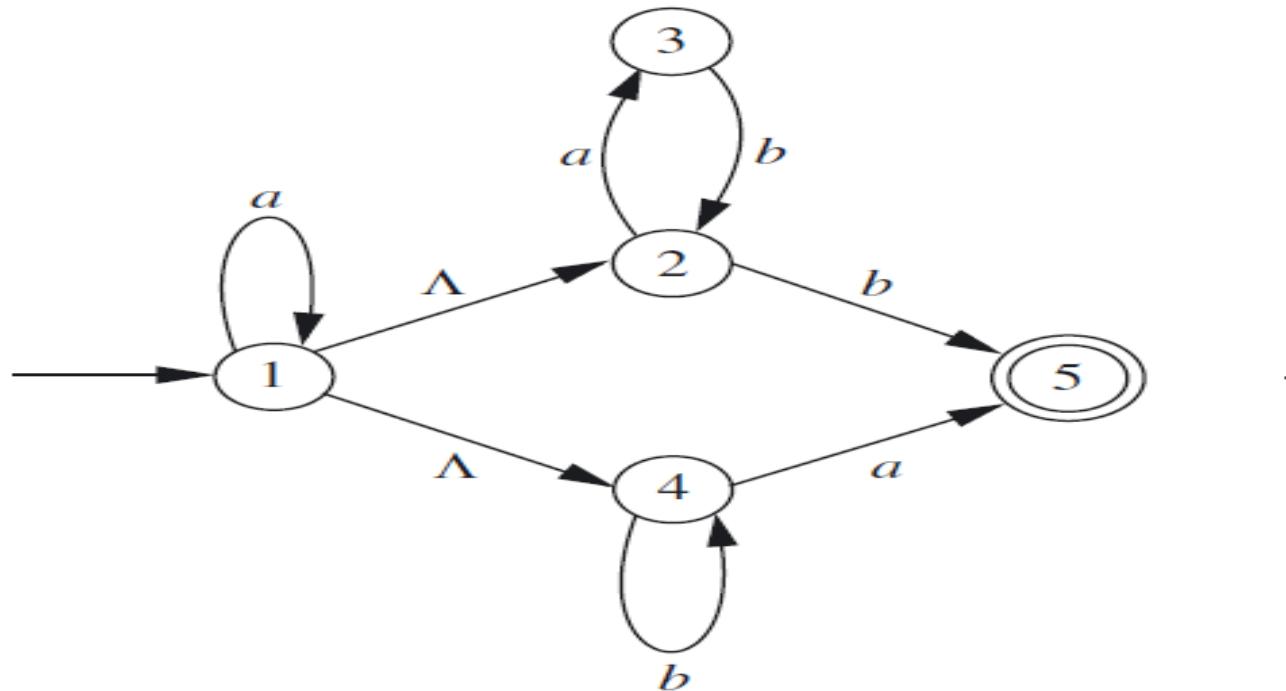
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$			
3						
4						
5						



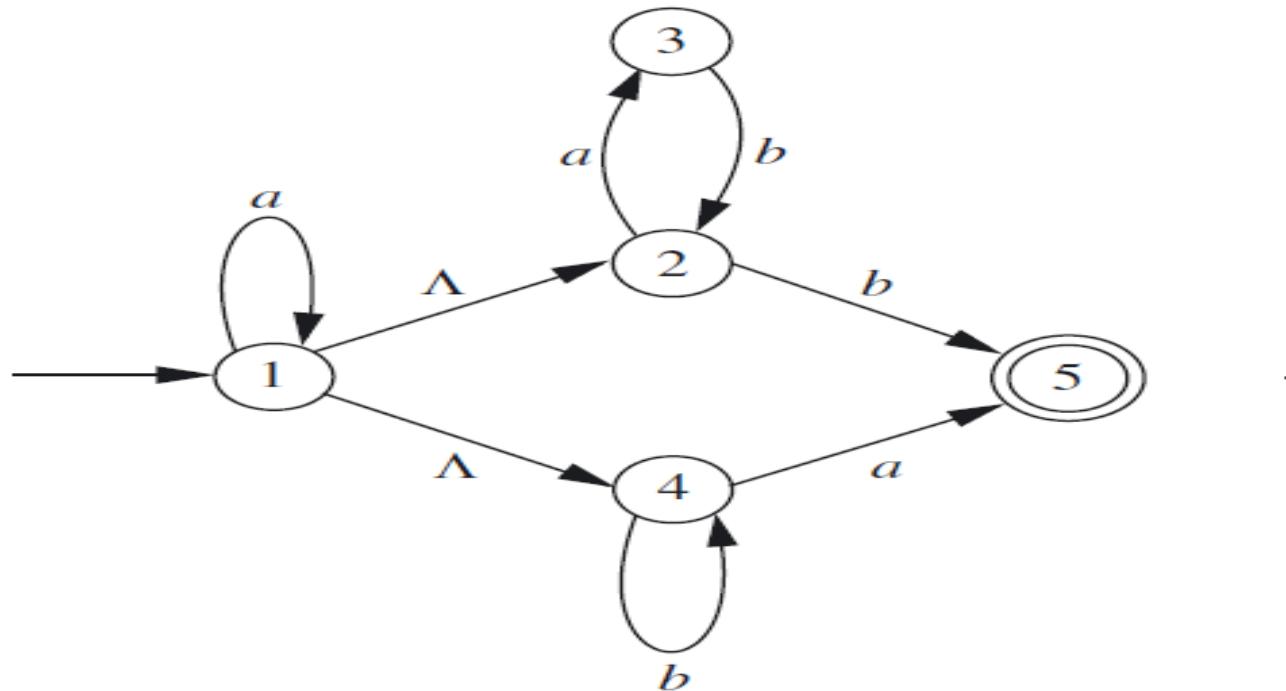
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$		
3						
4						
5						



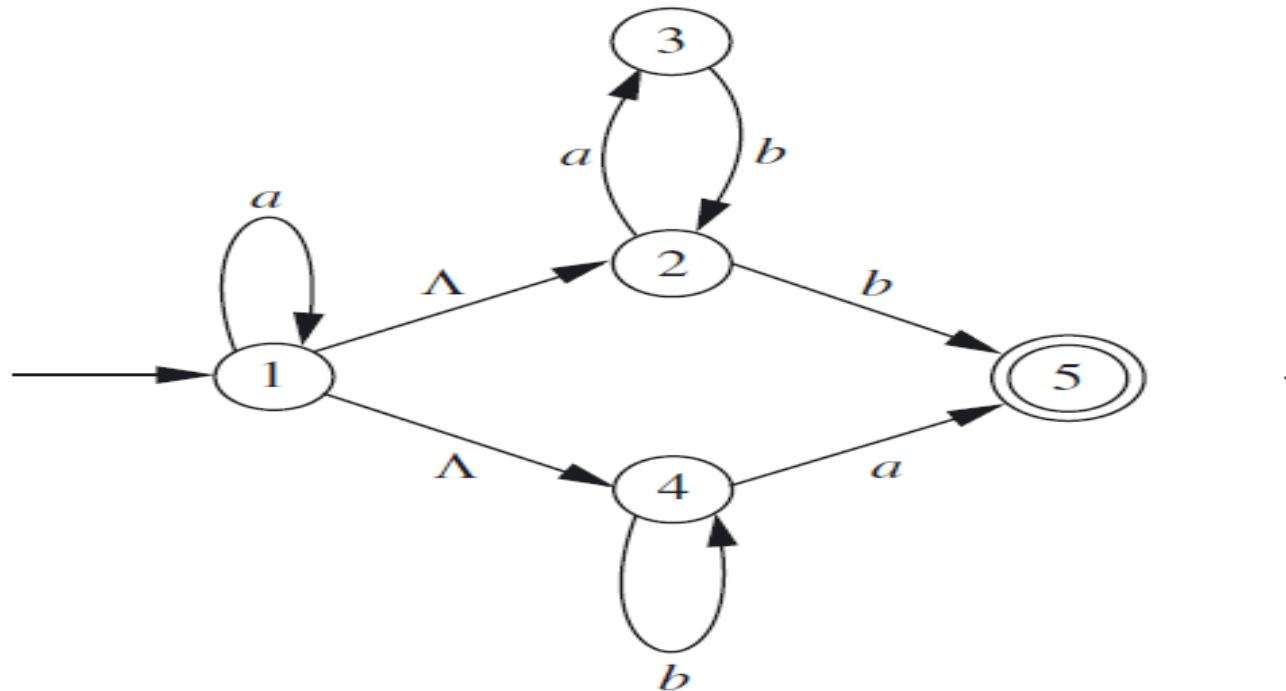
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	
3						
4						
5						



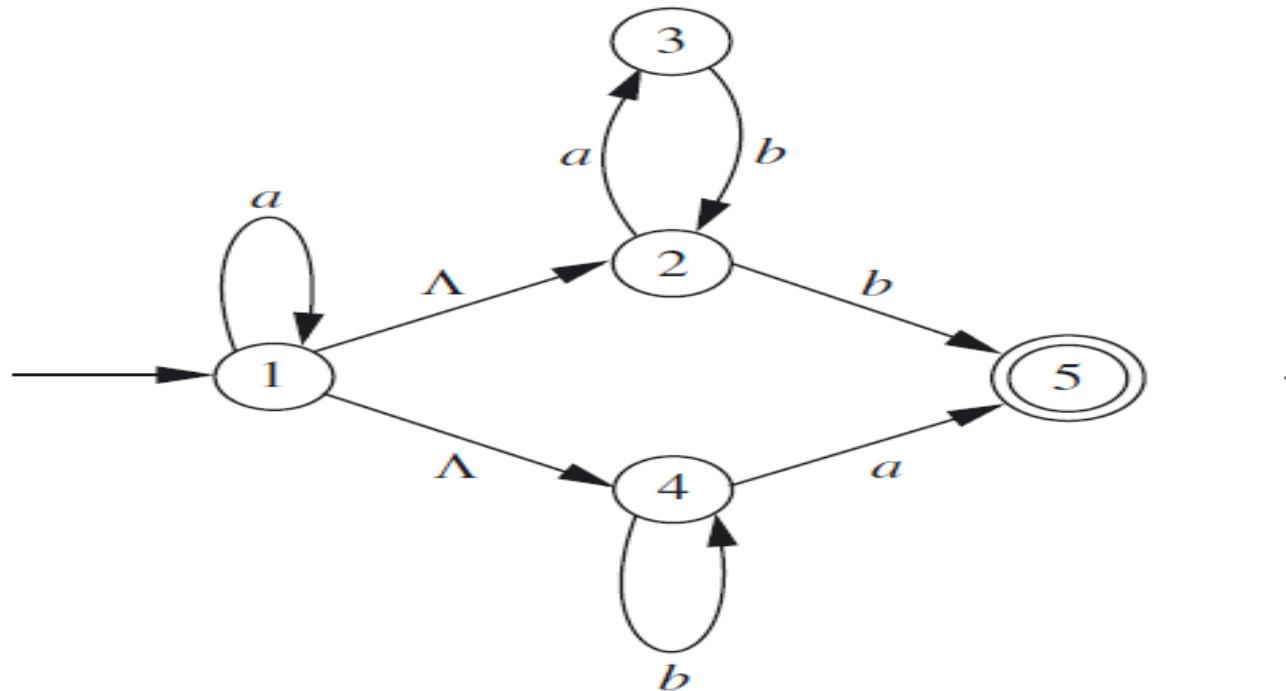
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3						
4						
5						



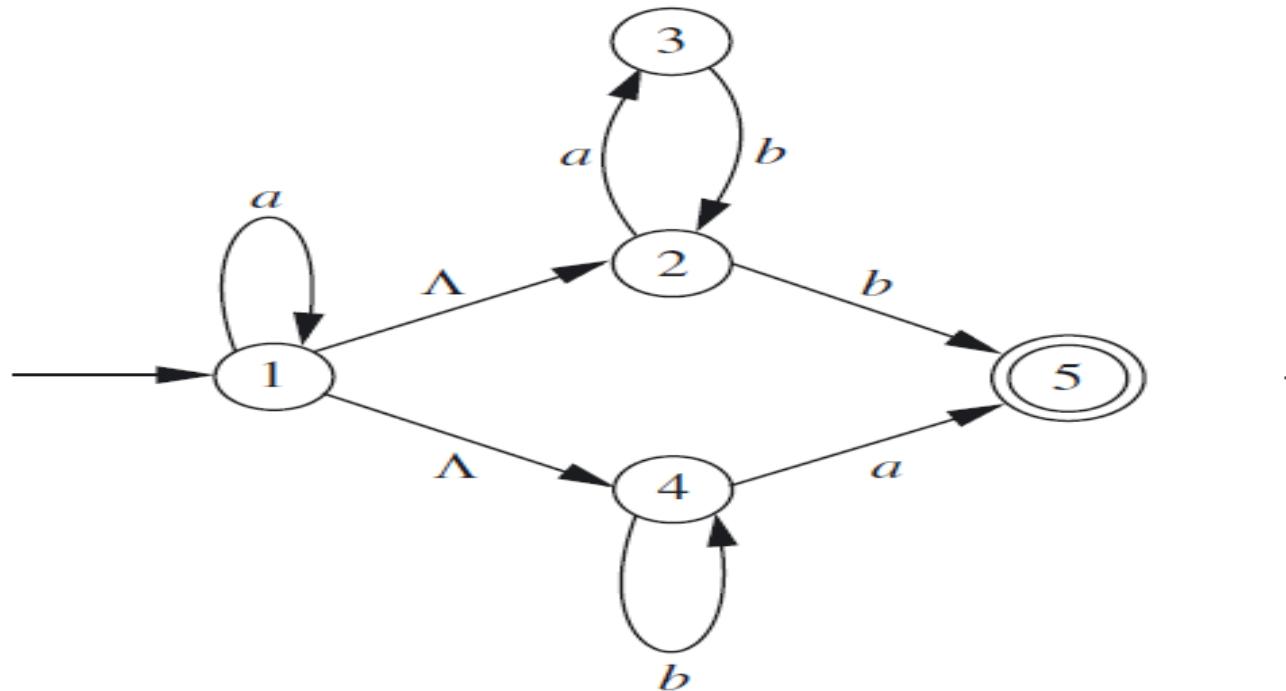
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$					
4						
5						



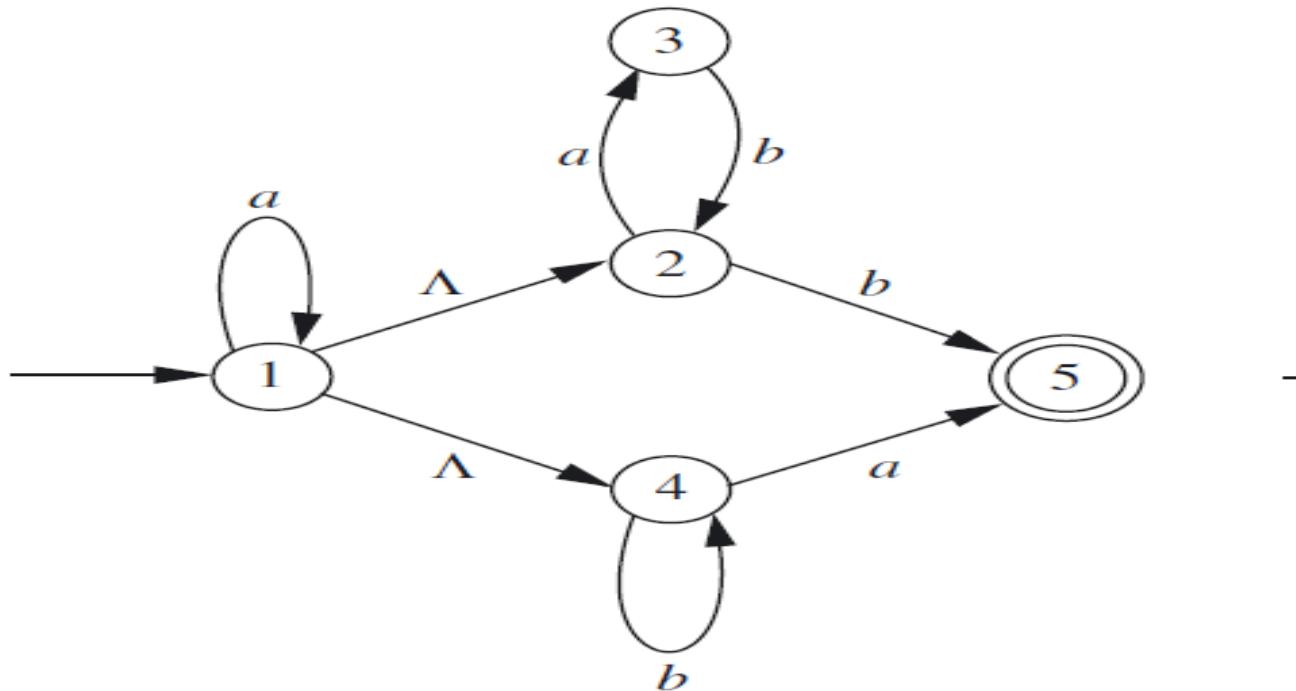
	$\lambda$	a			b	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$				
4						
5						



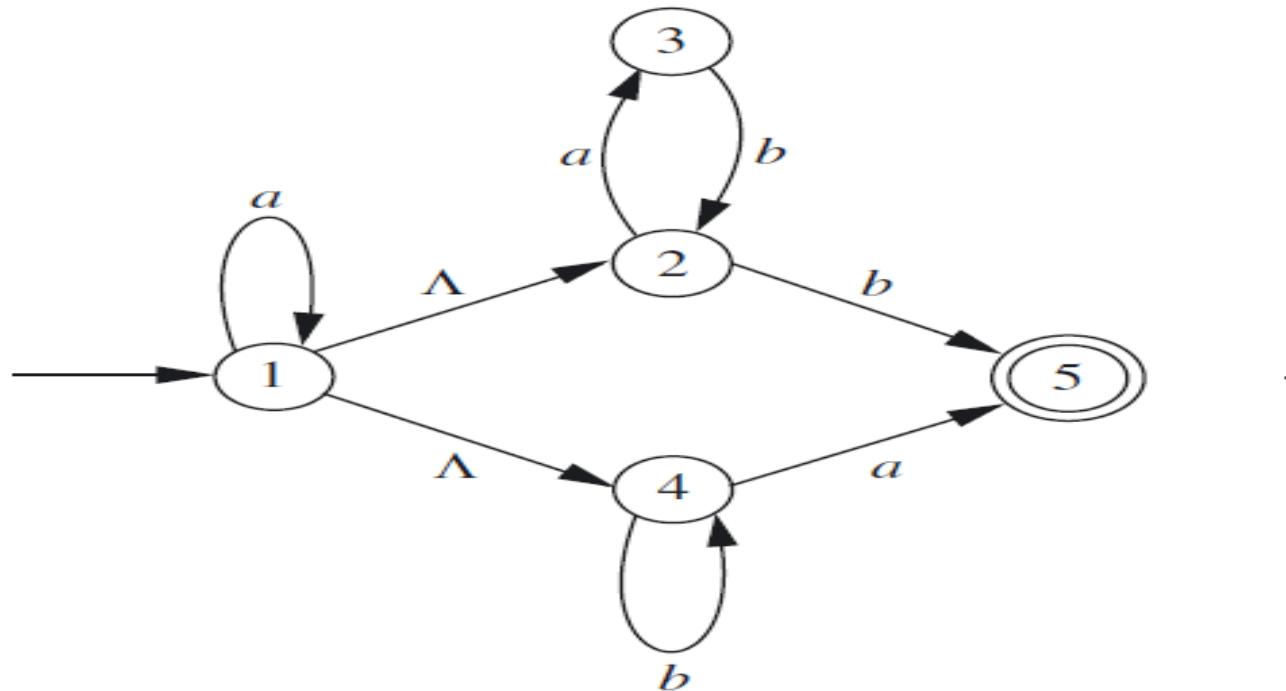
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$			
4						
5						



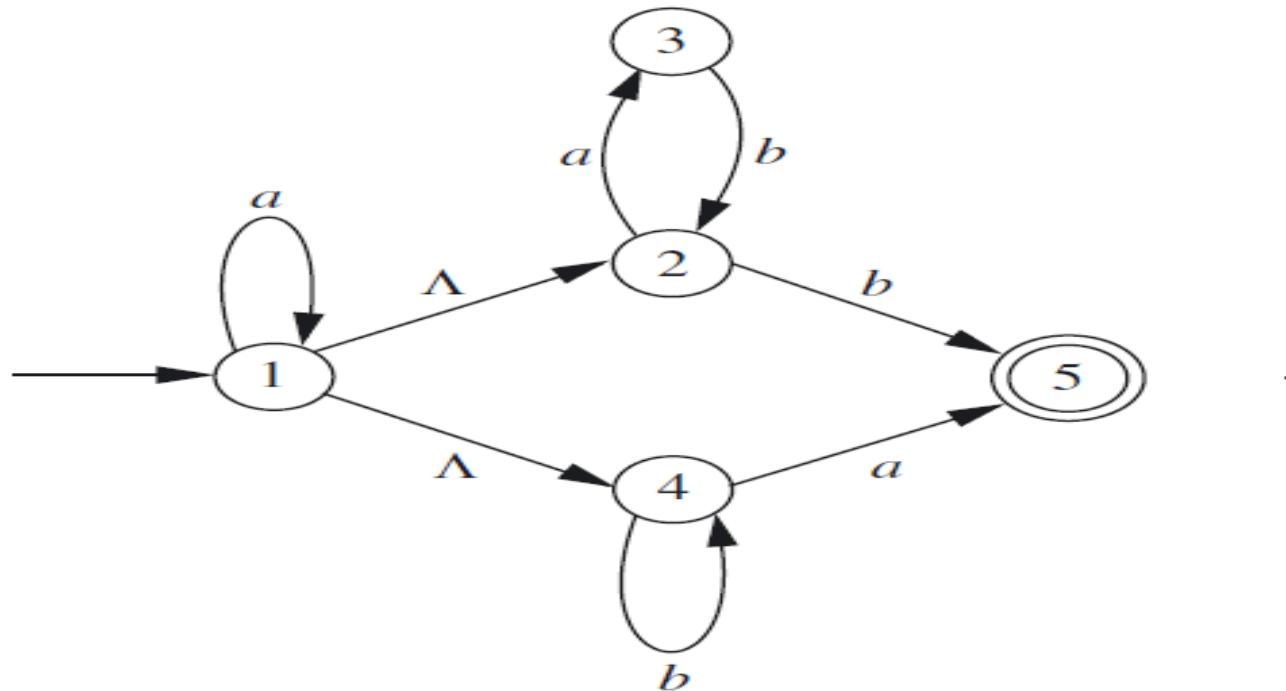
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$		
4						
5						



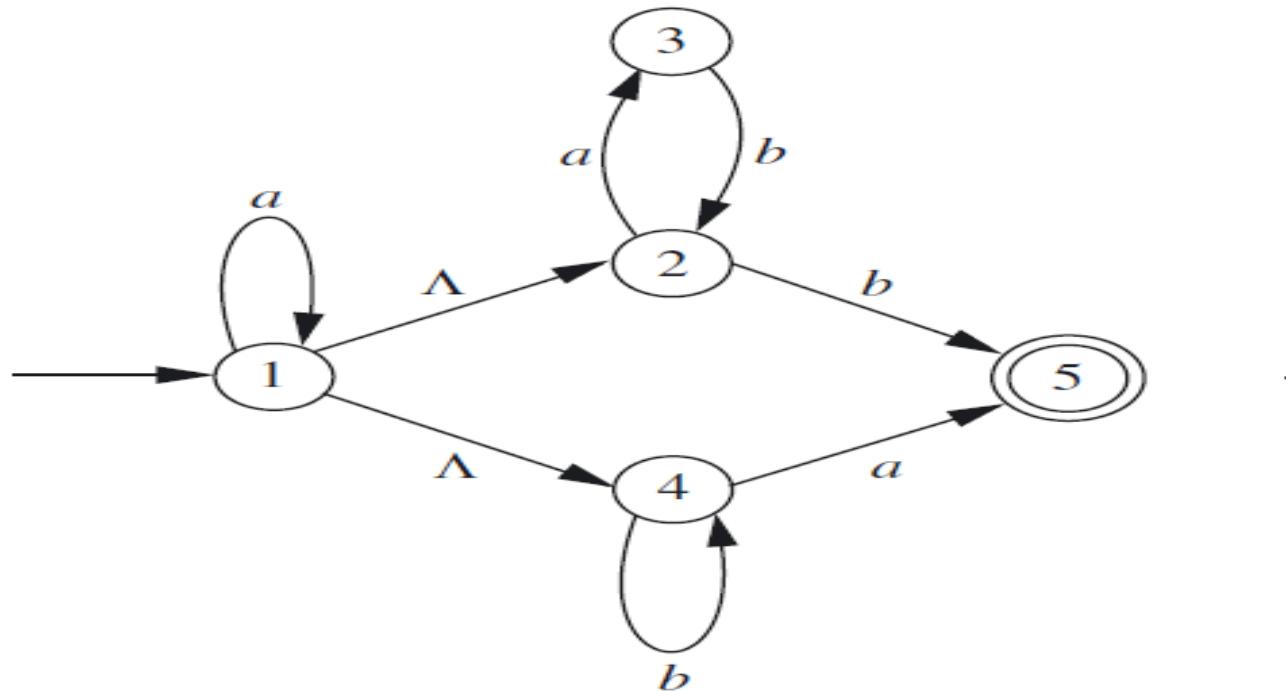
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	
4						
5						



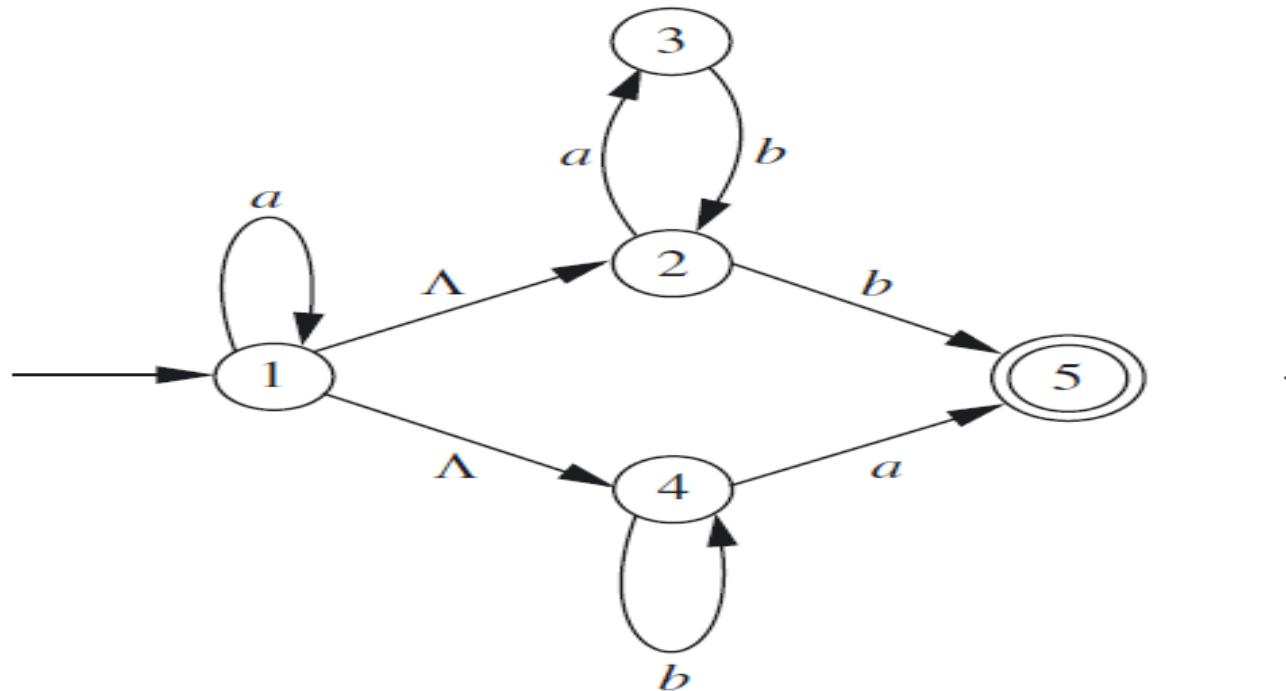
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4						
5						



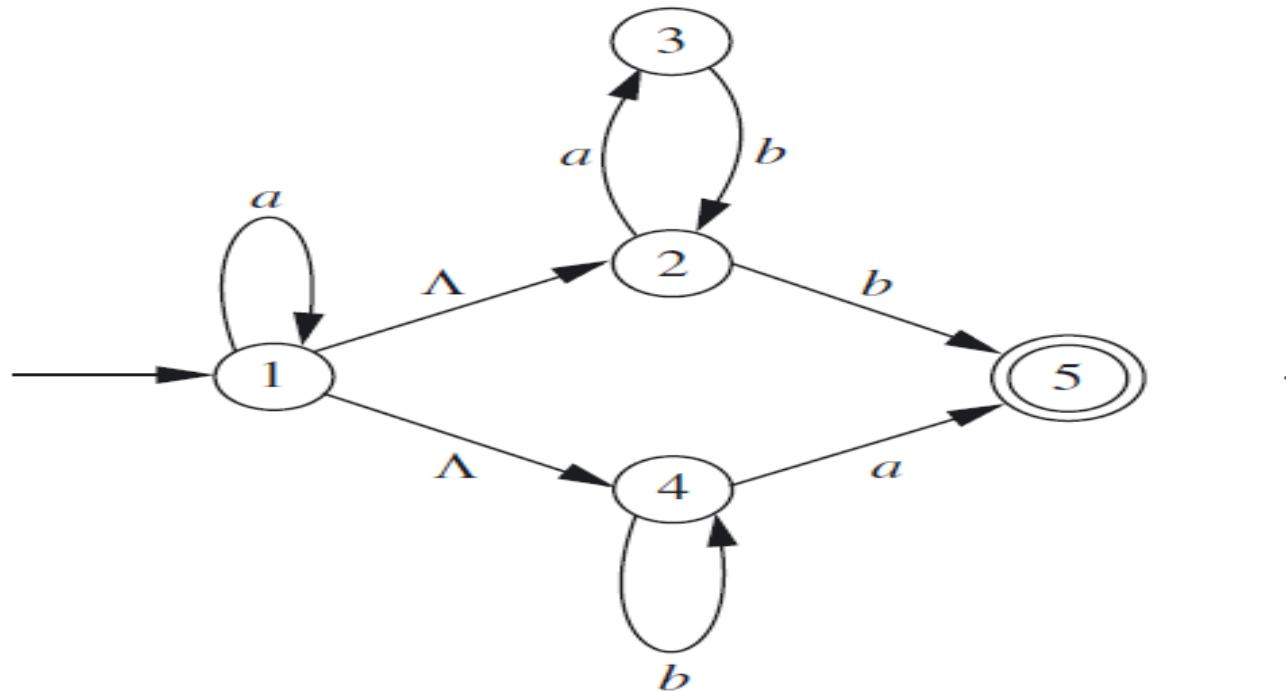
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$					
5						



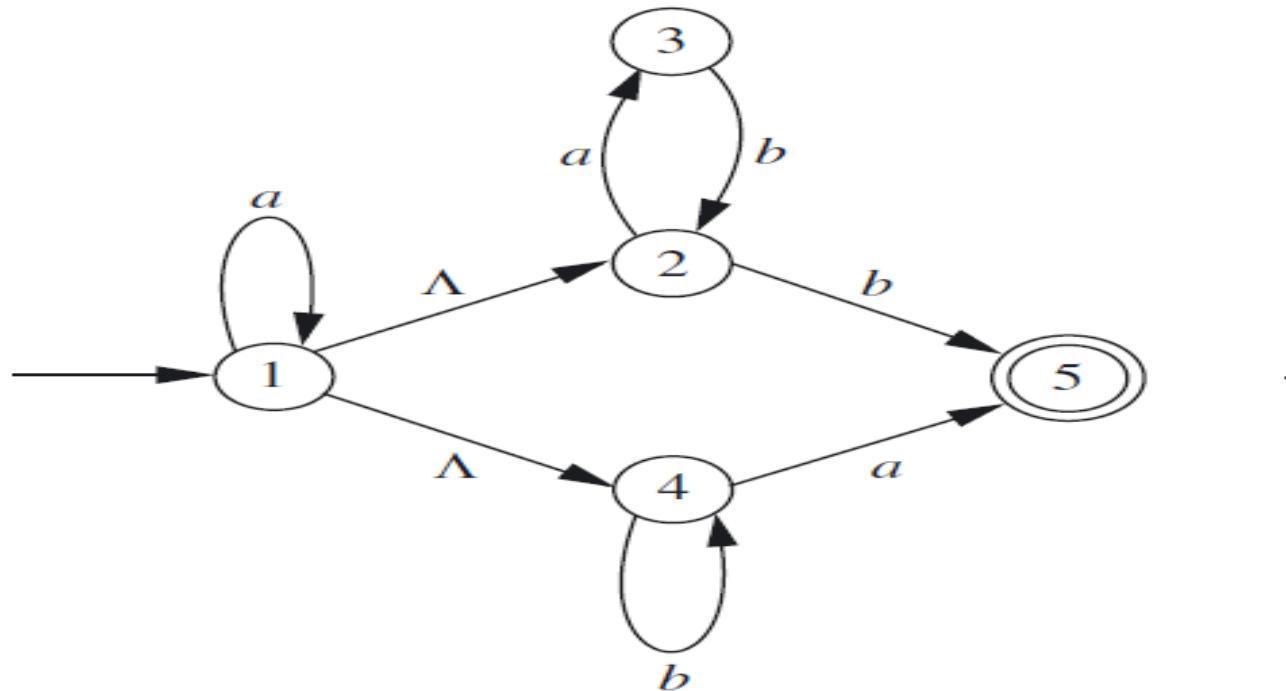
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$				
5						



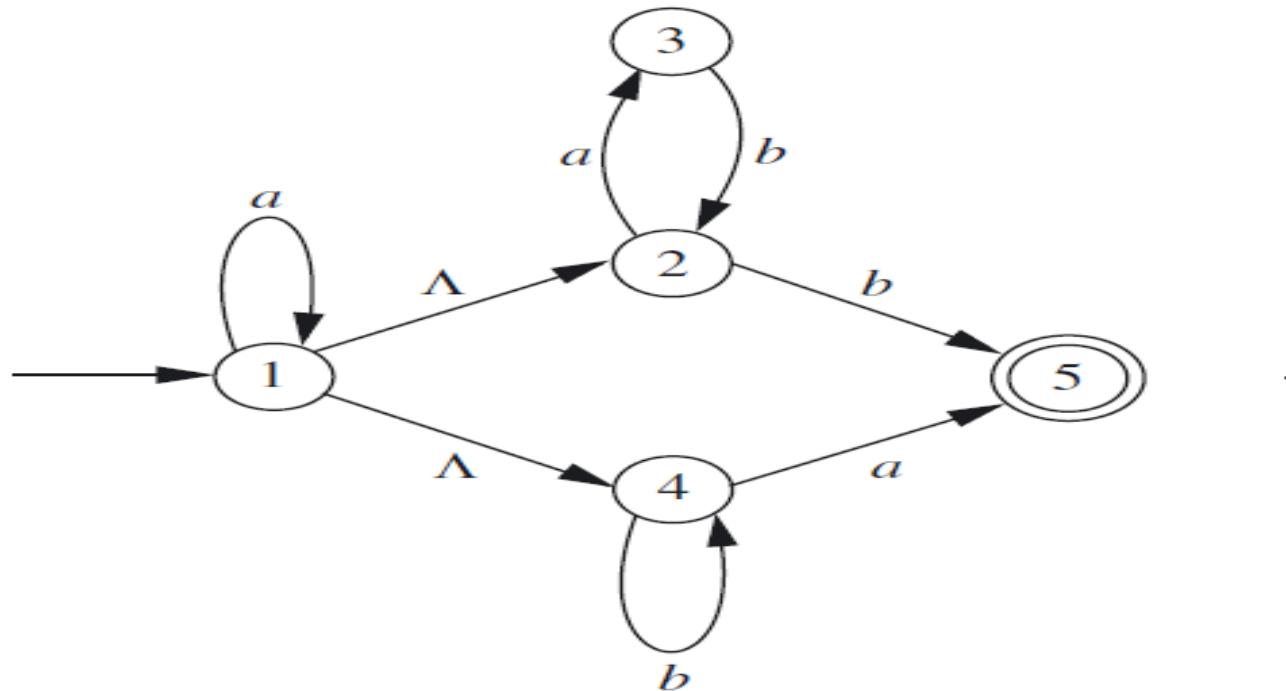
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$			
5						



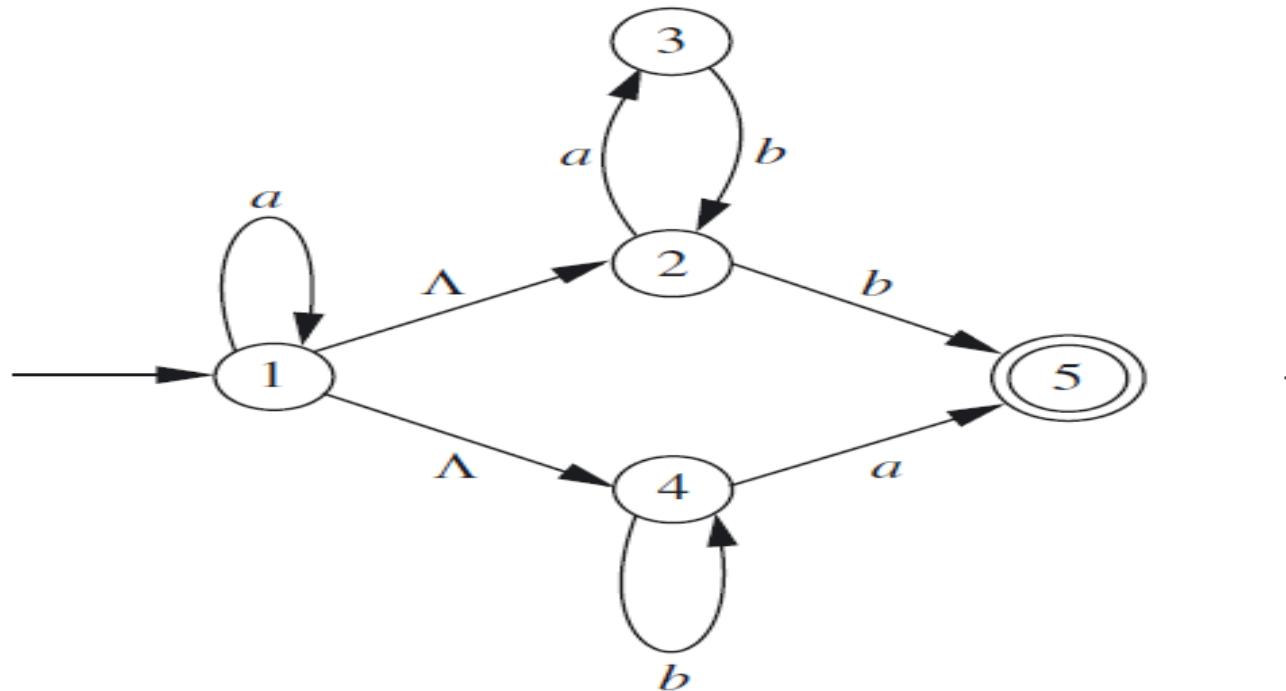
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$		
5						



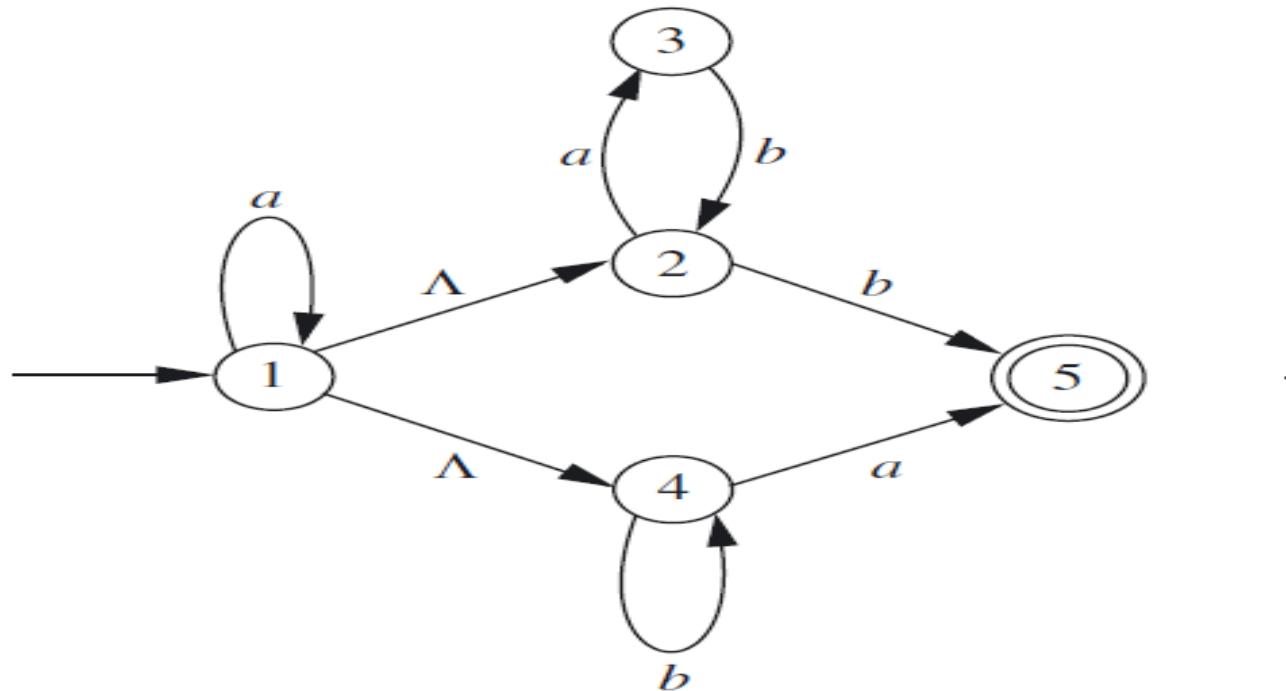
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	
5						



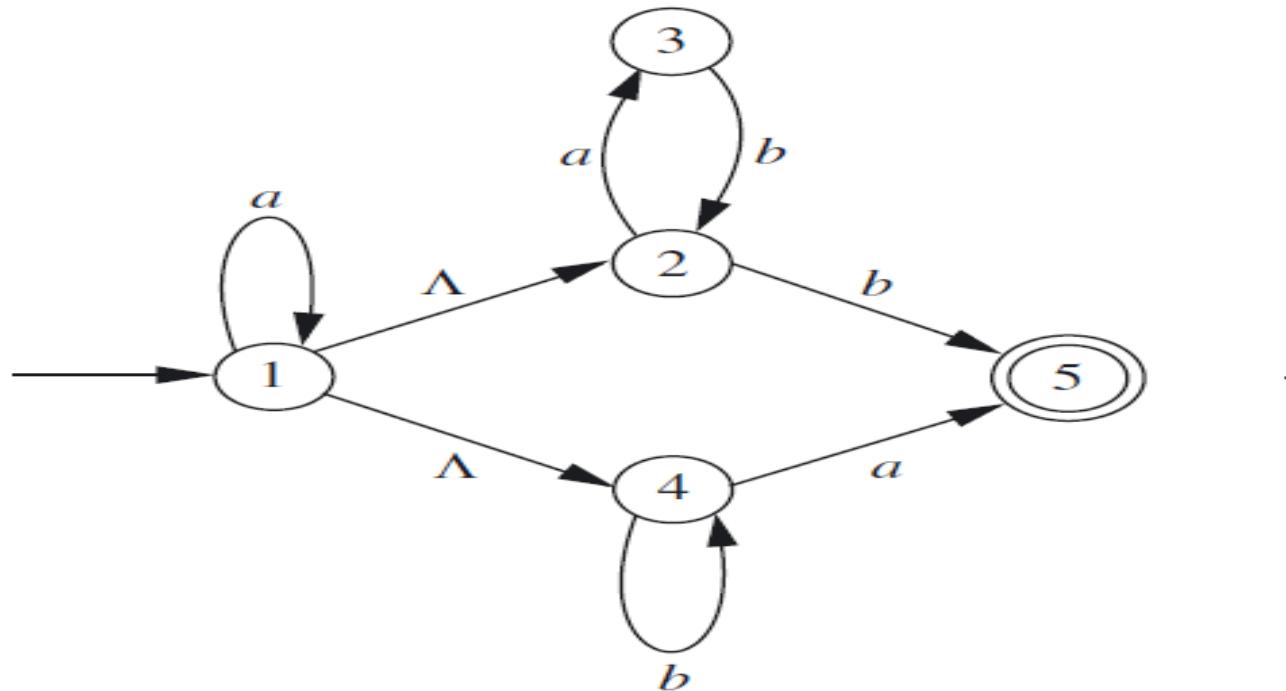
		a			b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5						



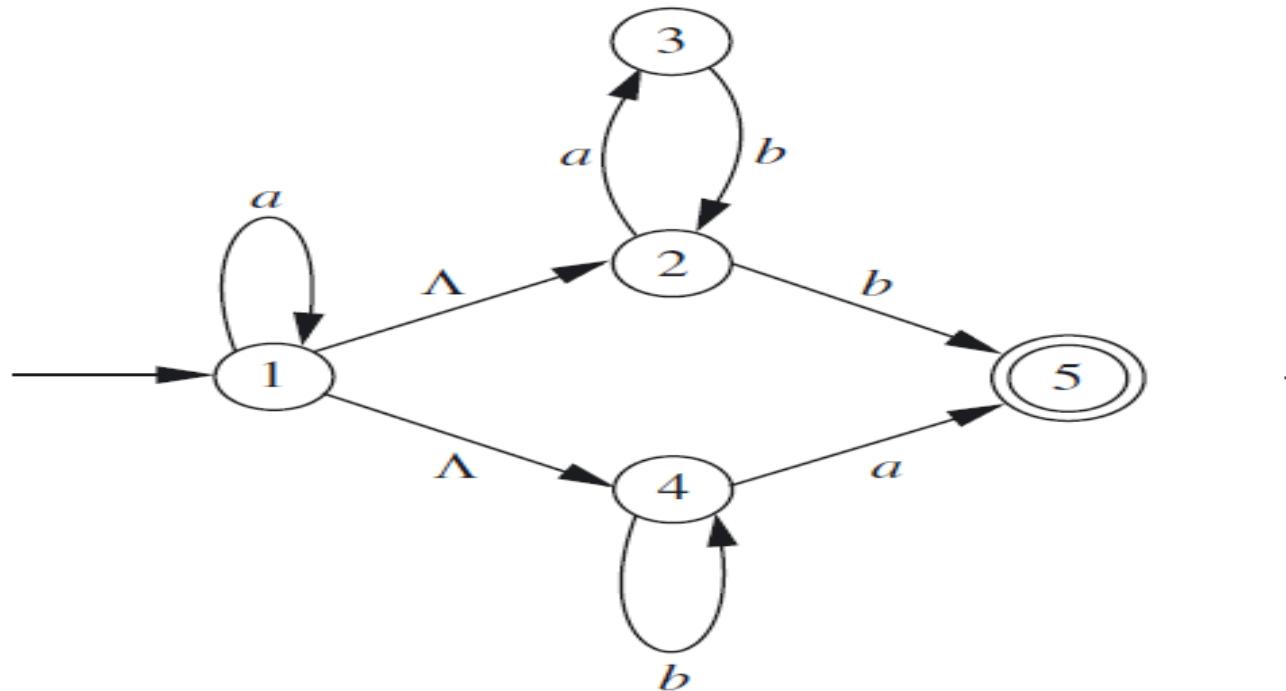
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$					



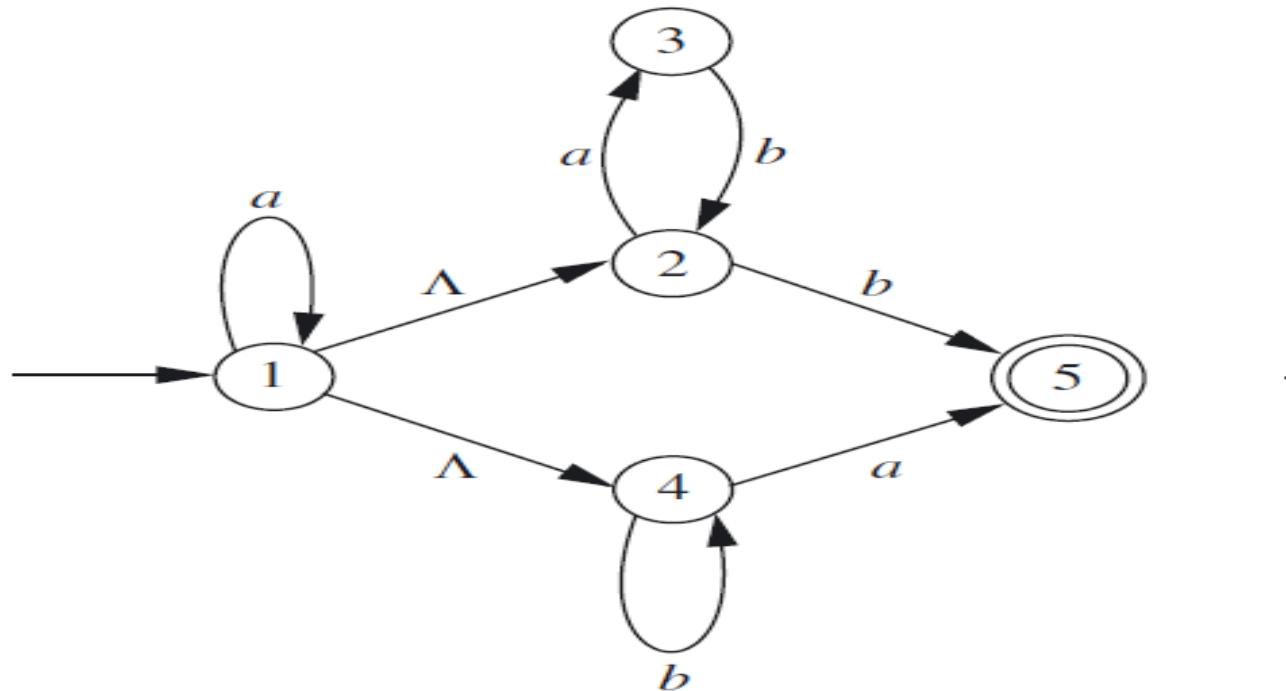
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$				



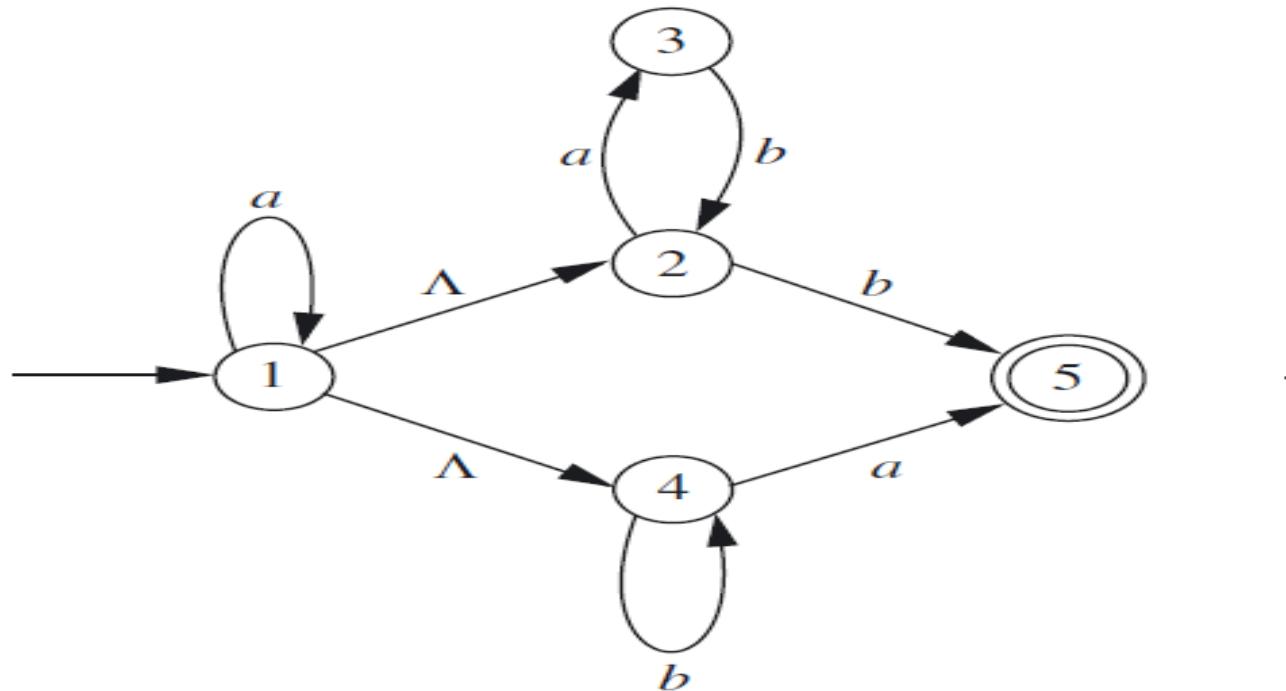
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$			



		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$		

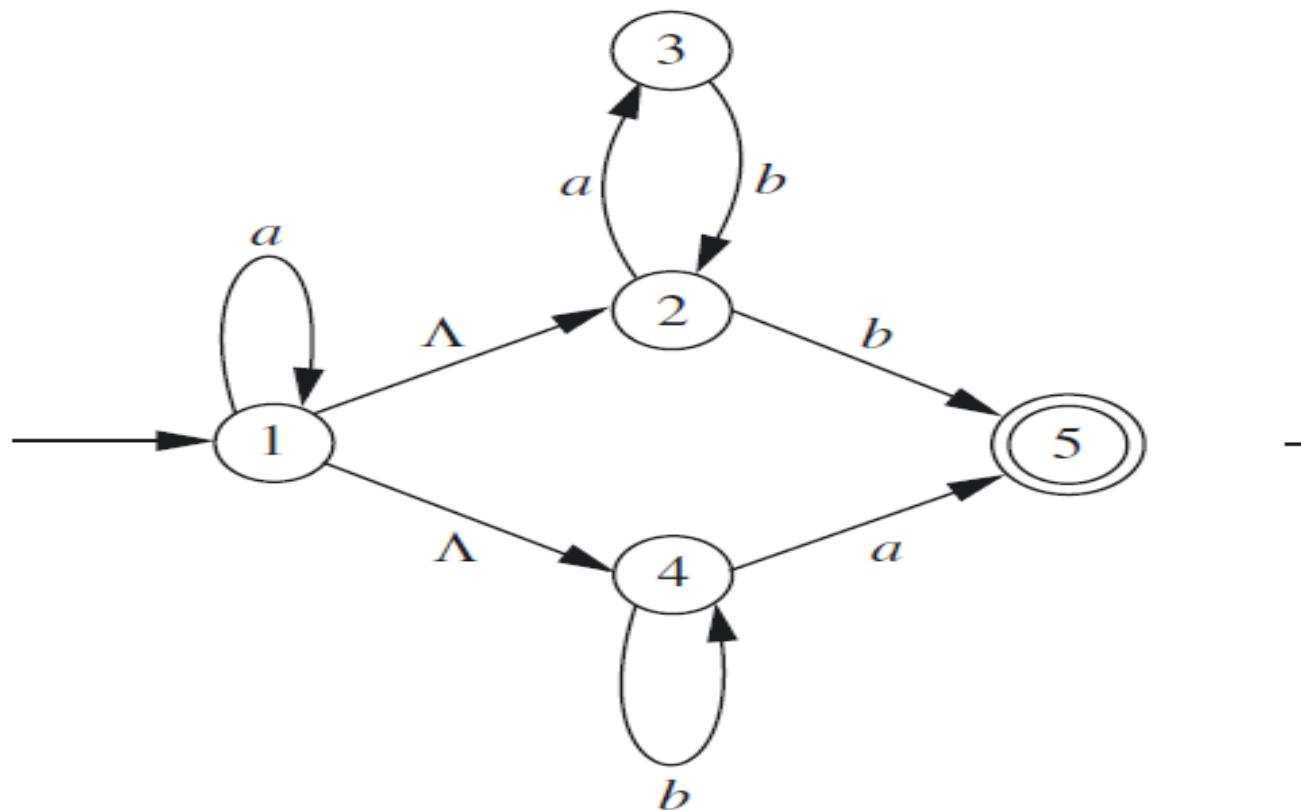


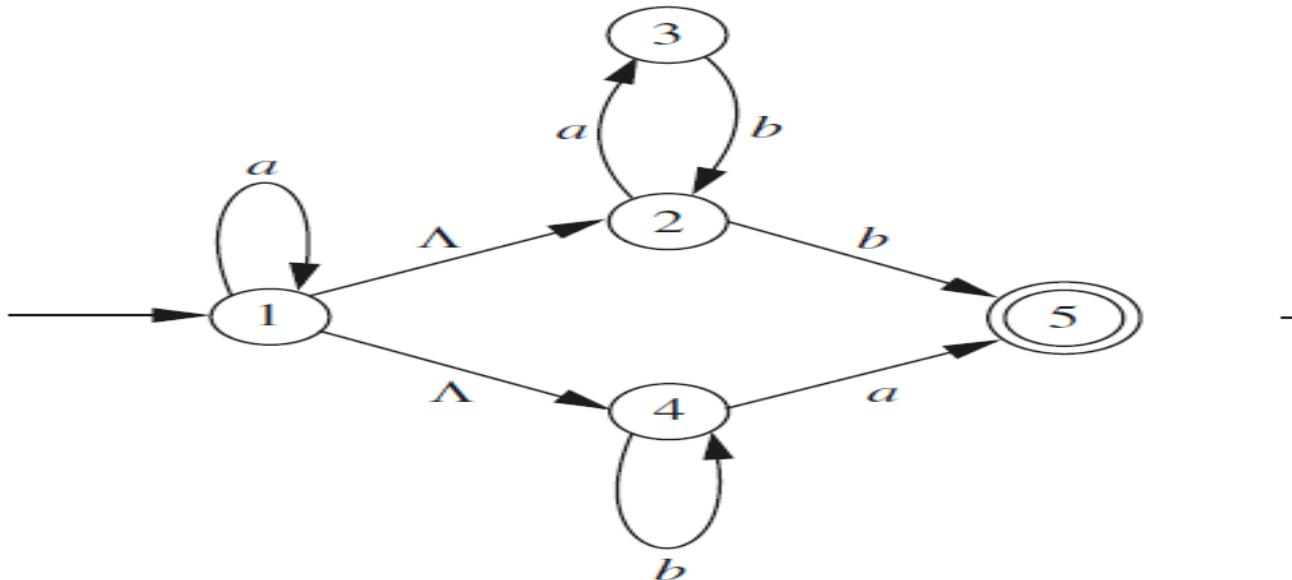
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	

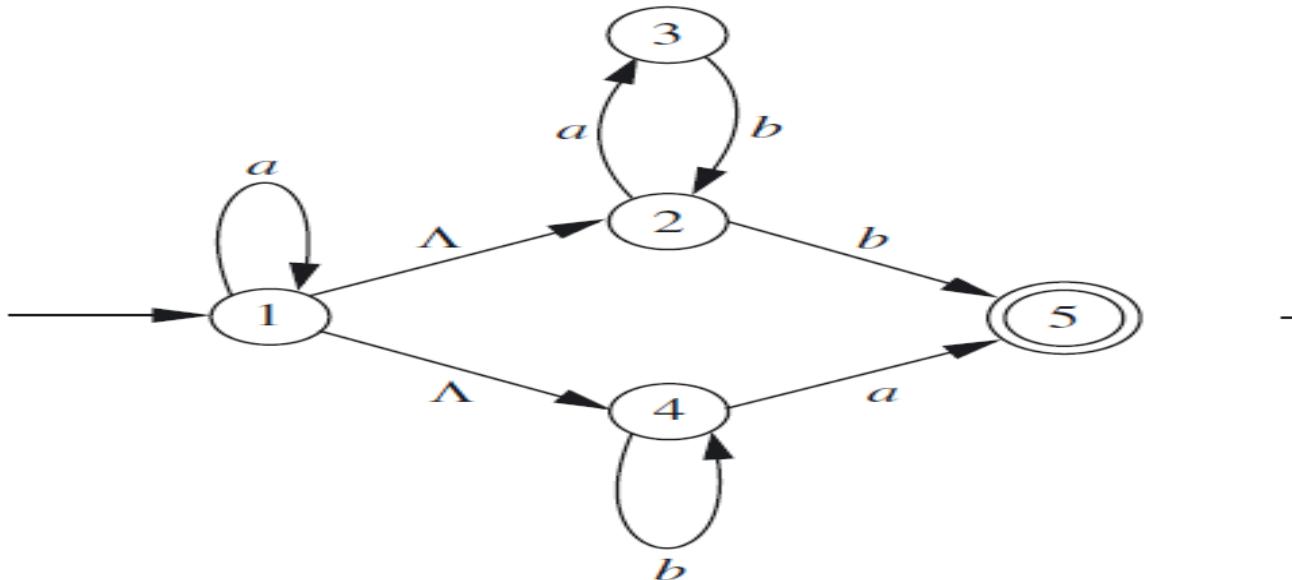


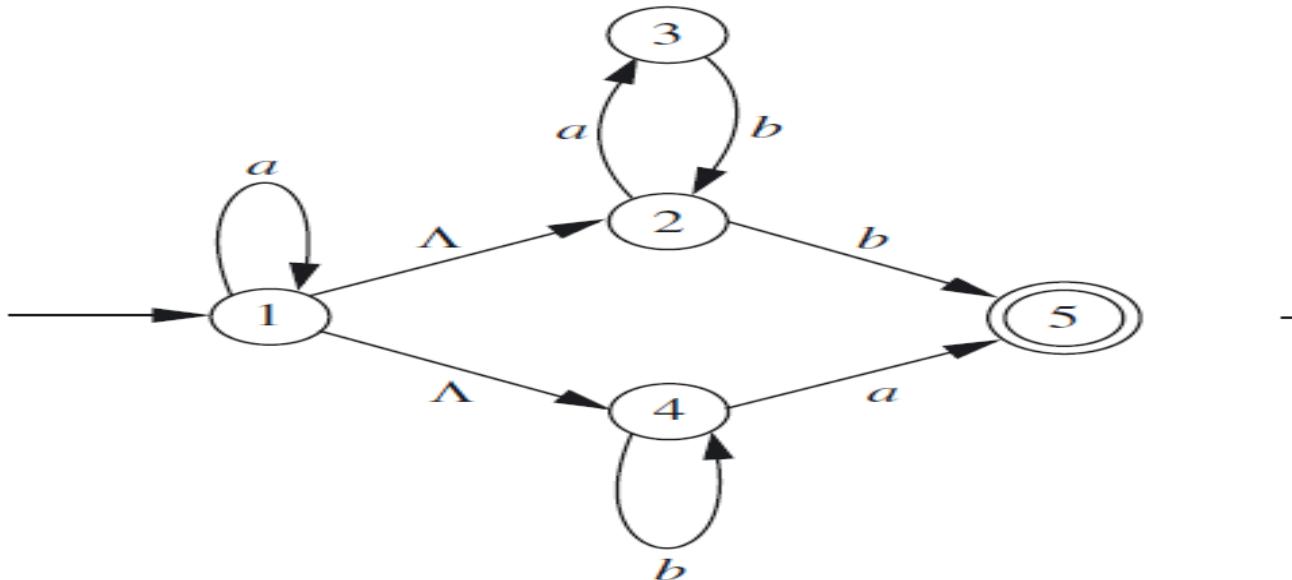
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$

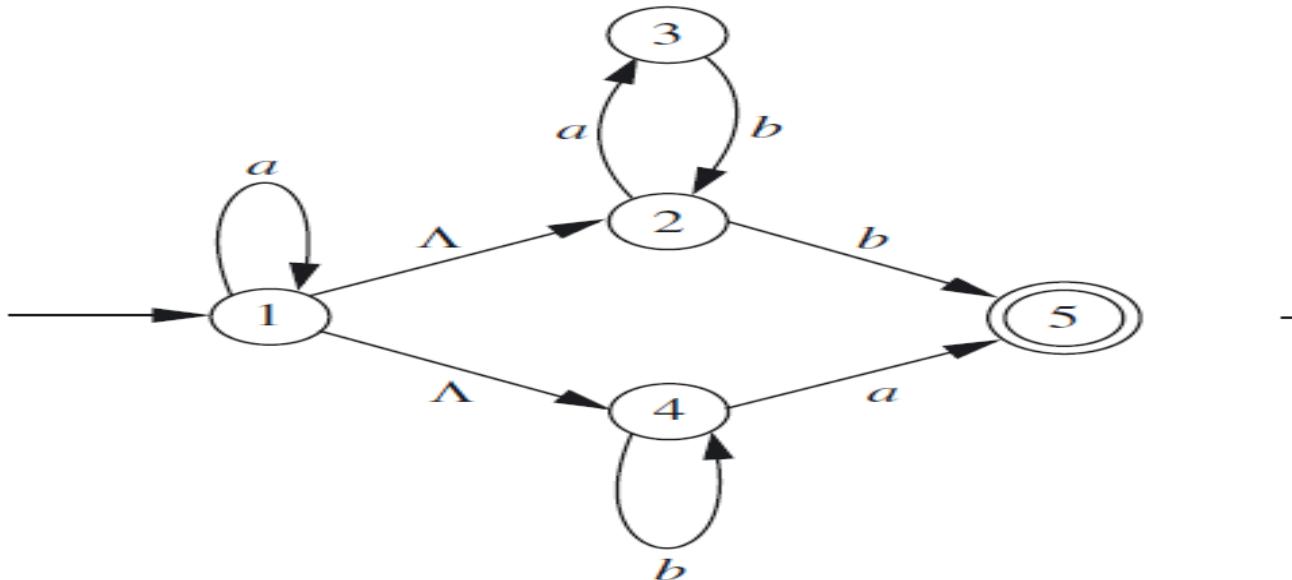
# Example for NFA-null to DFA Conversion

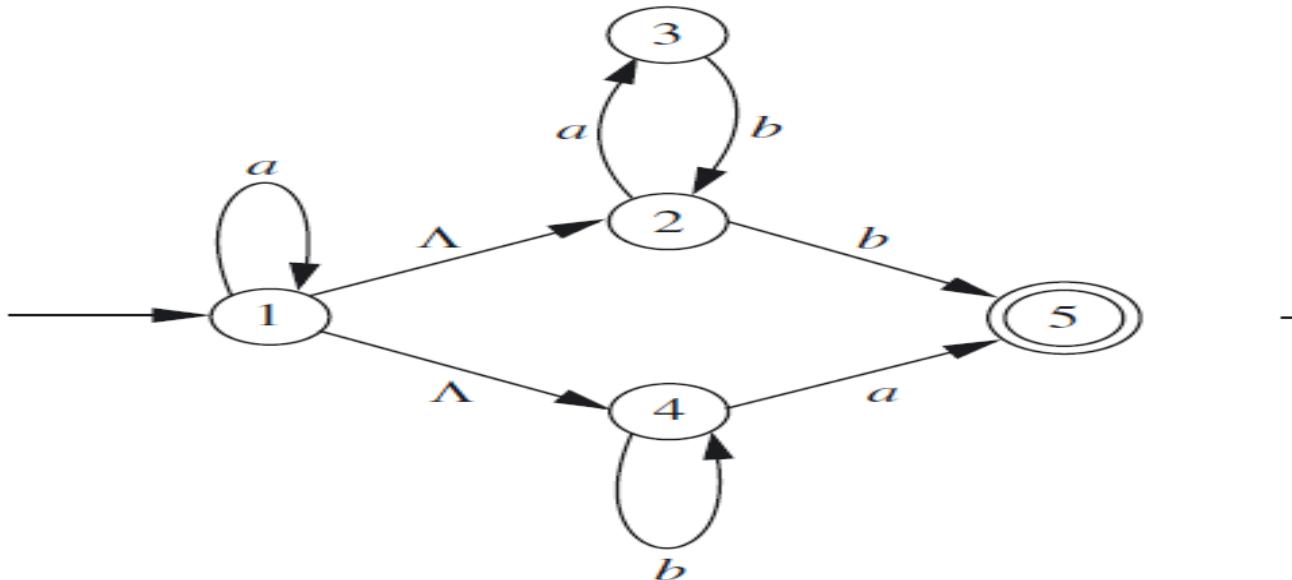


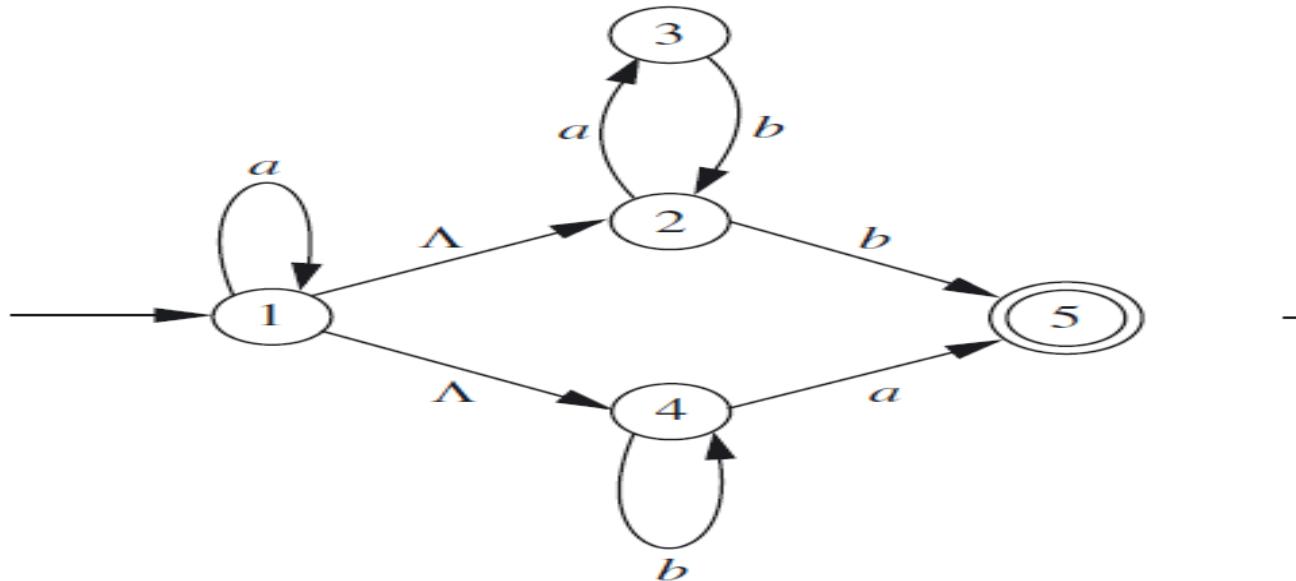




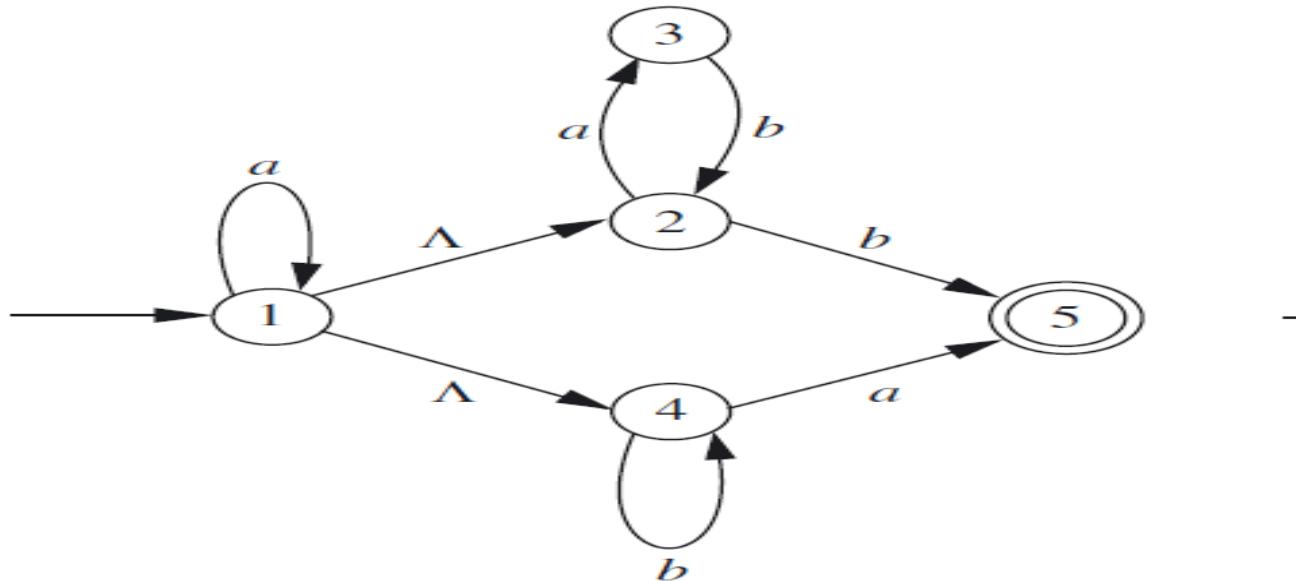




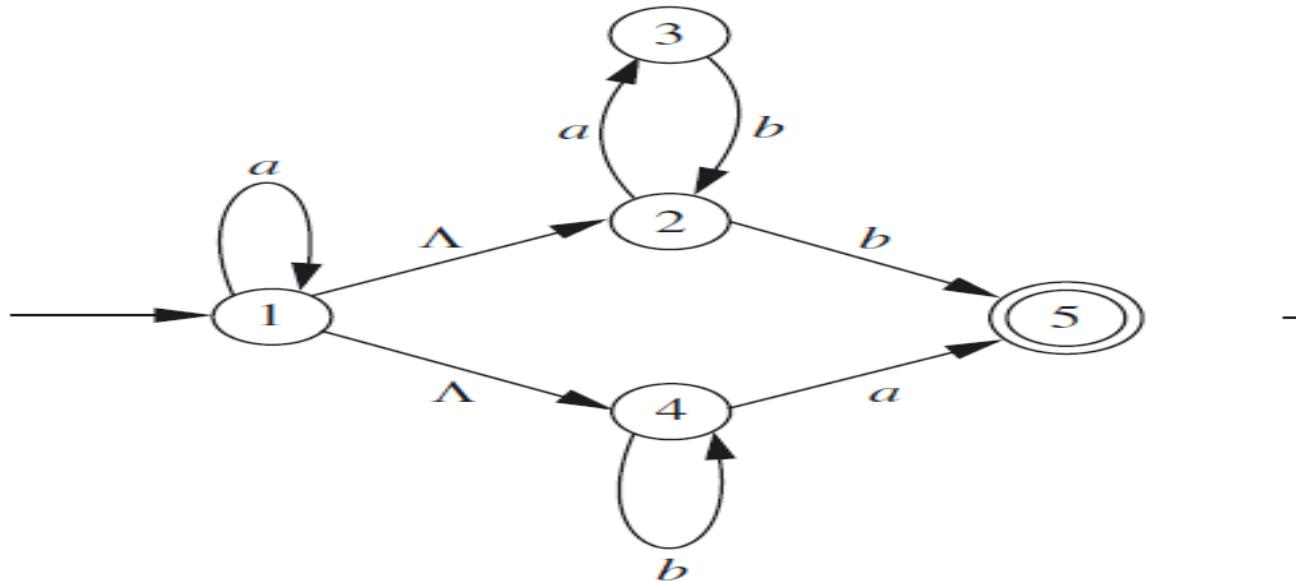




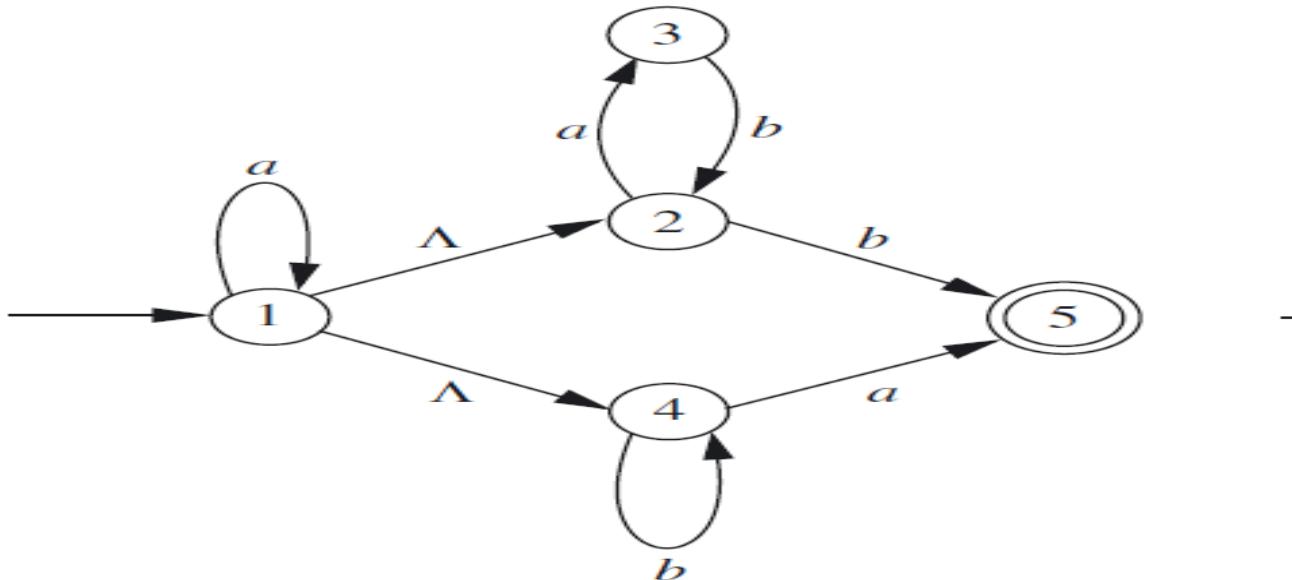
		$a$			$b$	
1	$\lambda$	$\{1,3,5\}$	$\lambda$	$\{1,2,3,4,5\}$	$\lambda$	$\{4,5\}$

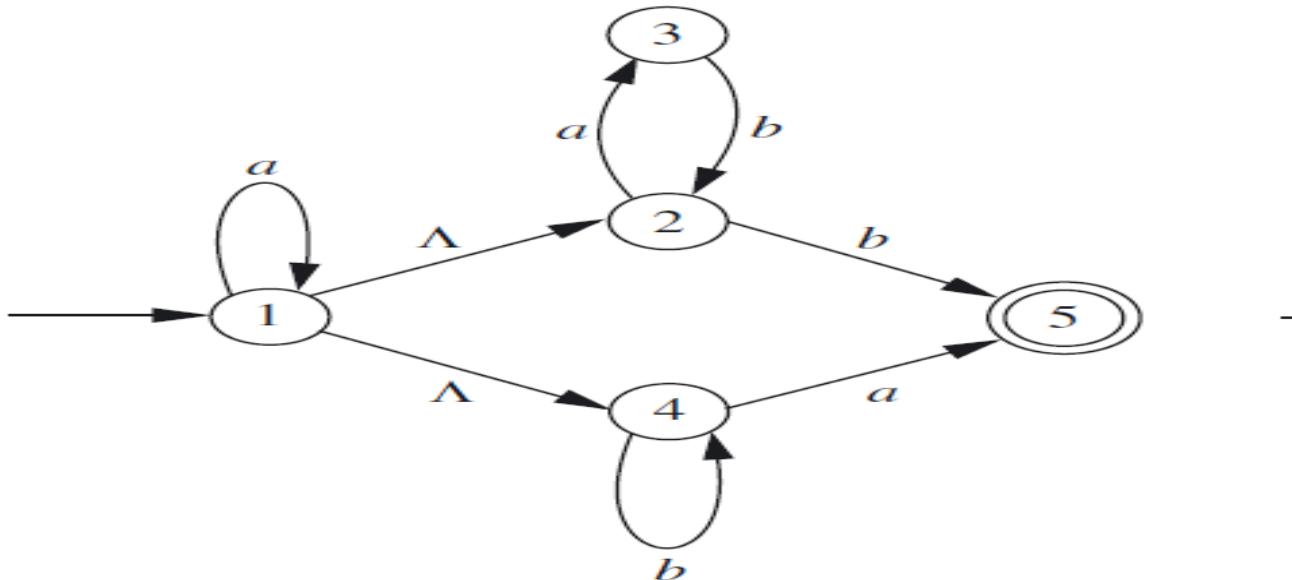


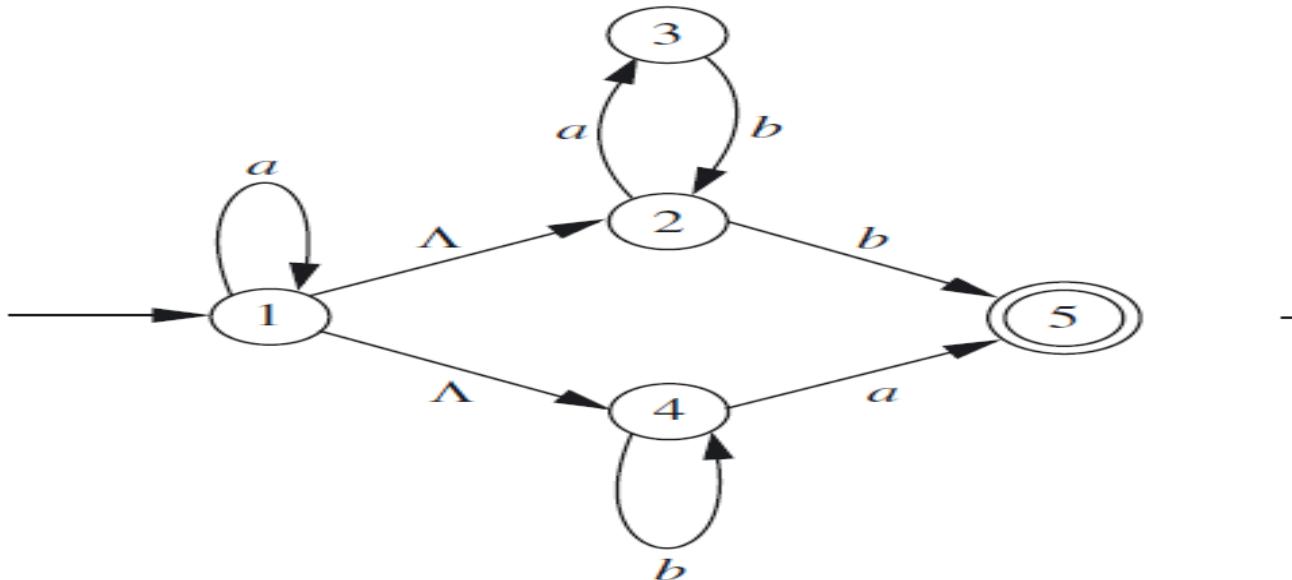
		$a$			$b$	
1	$\lambda$	$\{1,3,5\}$	$\lambda$	$\{1,2,3,4,5\}$	$\lambda$	$\{4,5\}$

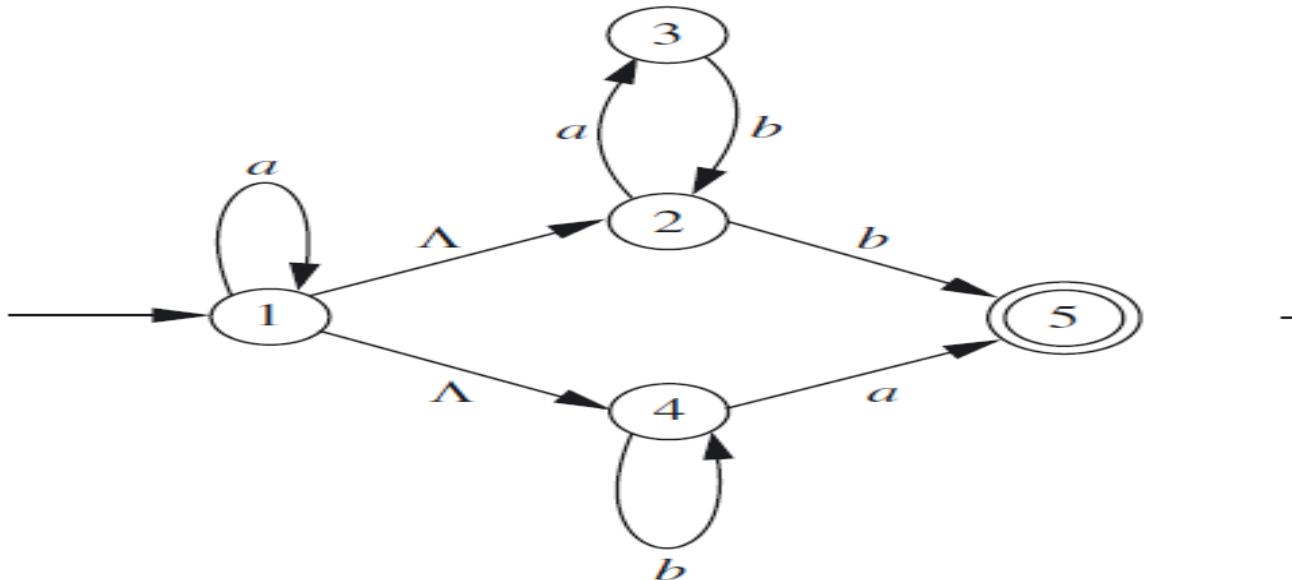


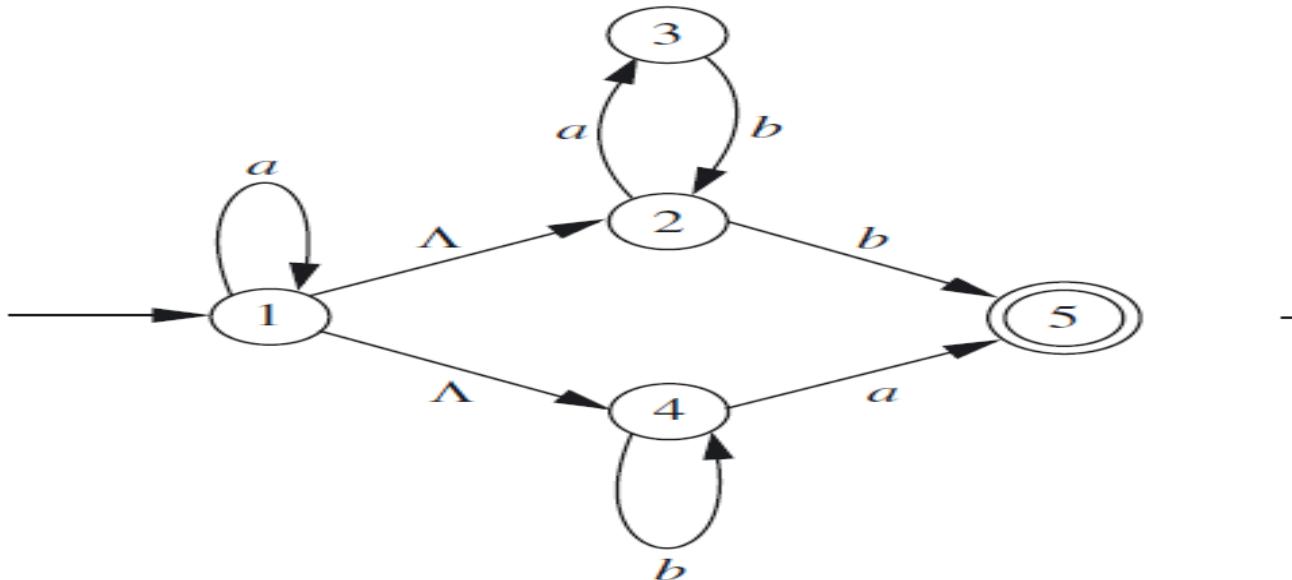
		$a$			$b$	
1	$\lambda$	$\{1,3,5\}$	$\lambda$	$\{1,2,3,4,5\}$	$\lambda$	$\{4,5\}$

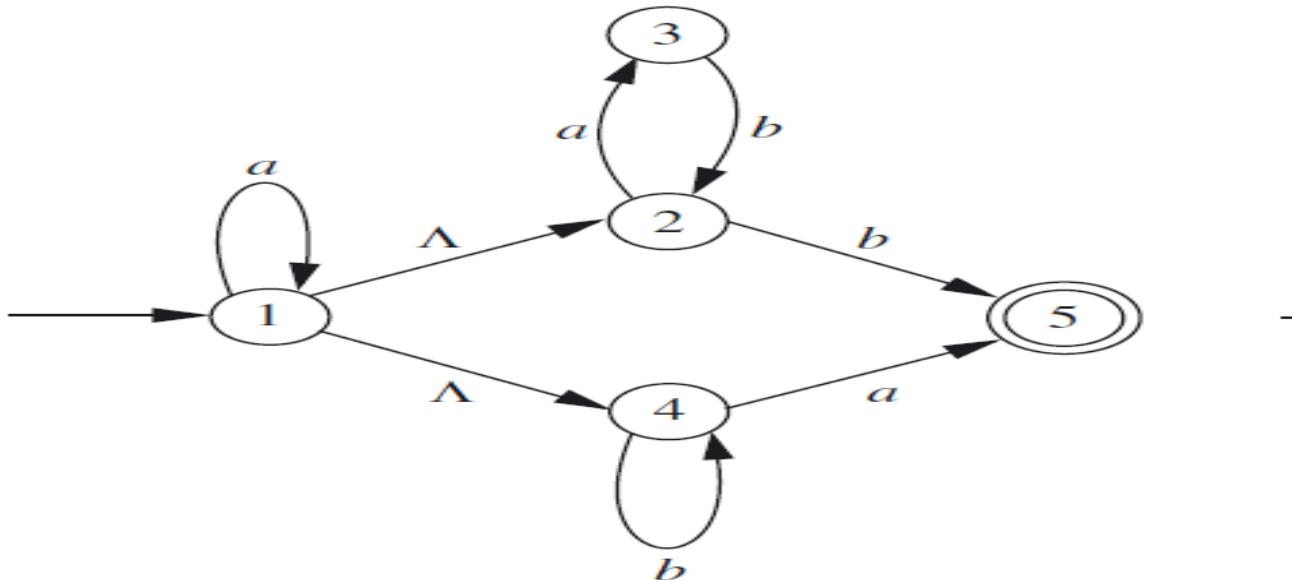


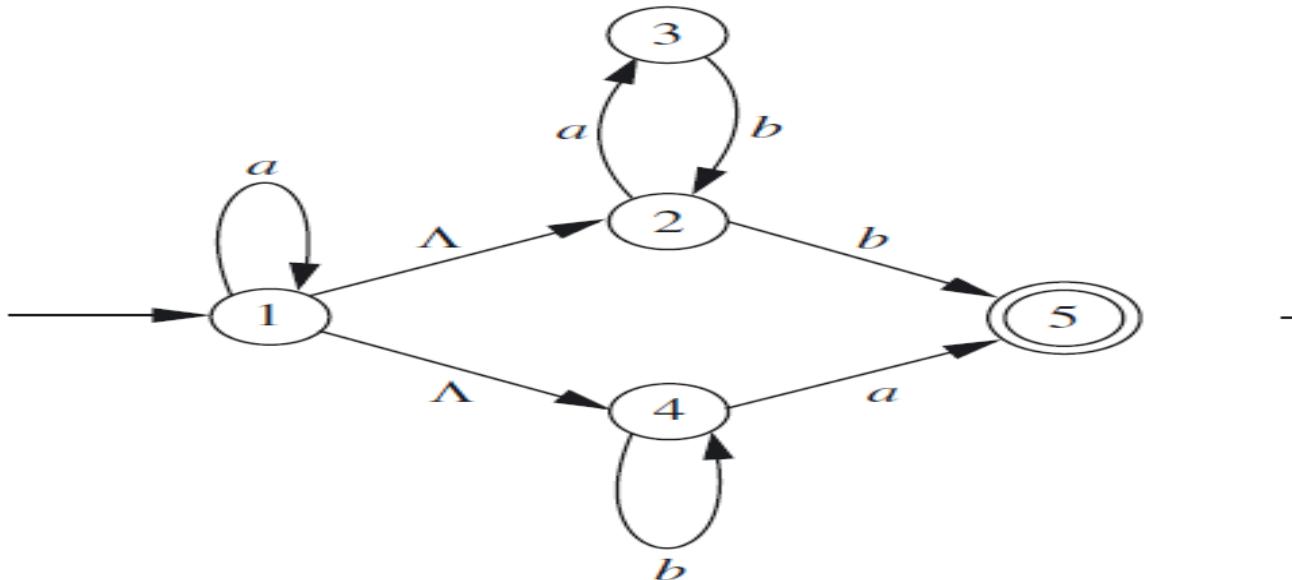




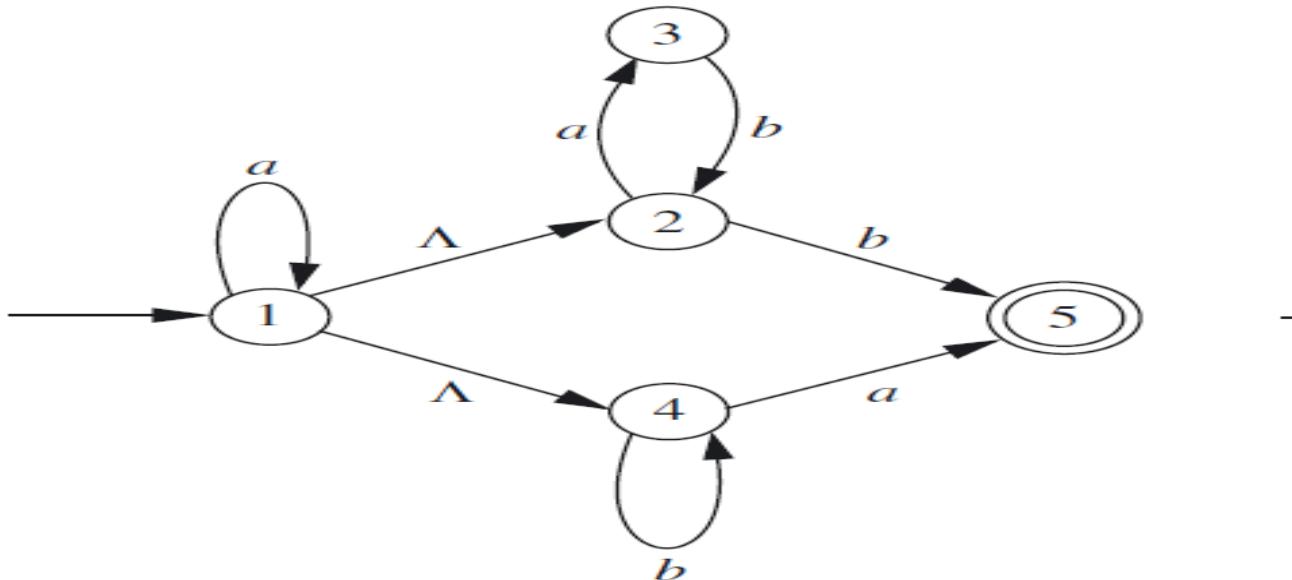


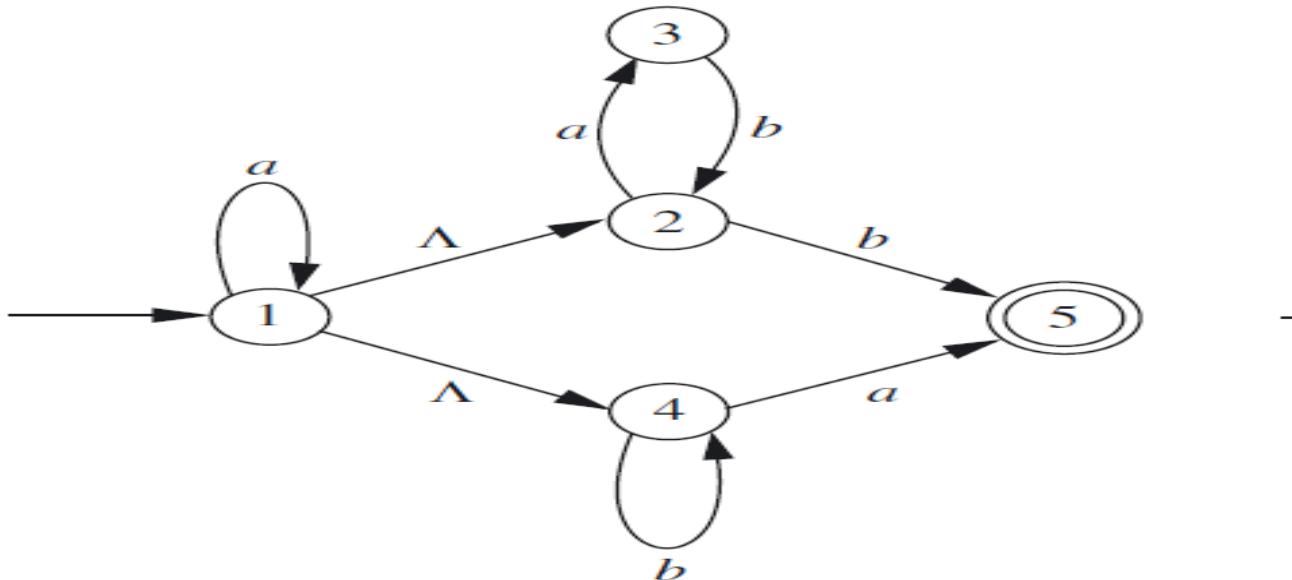


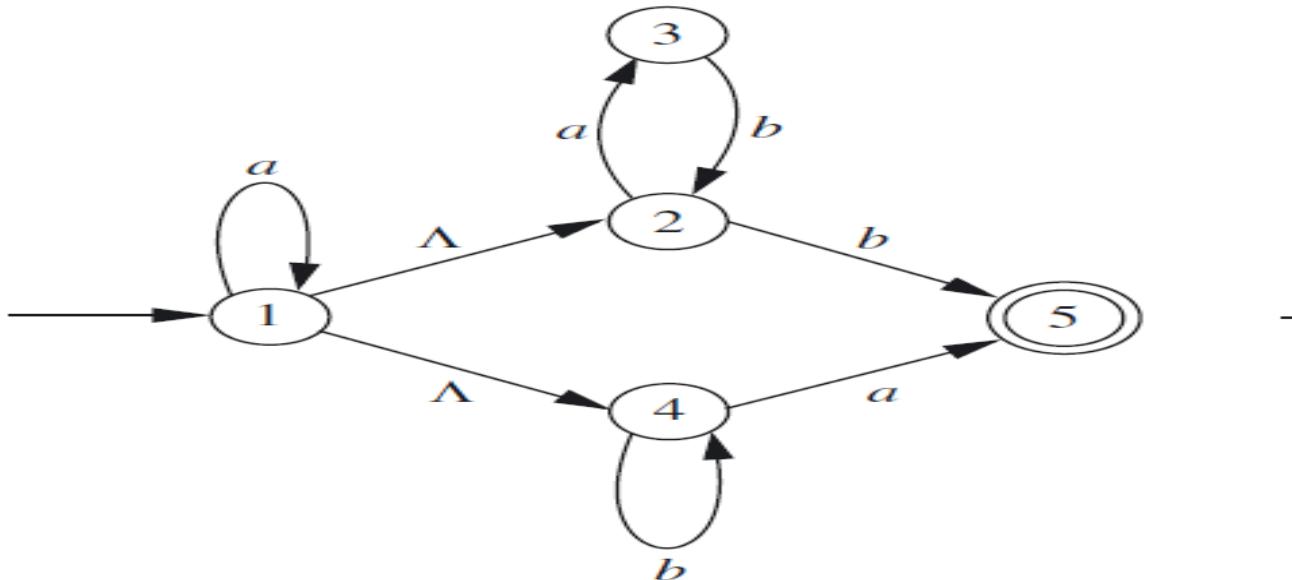


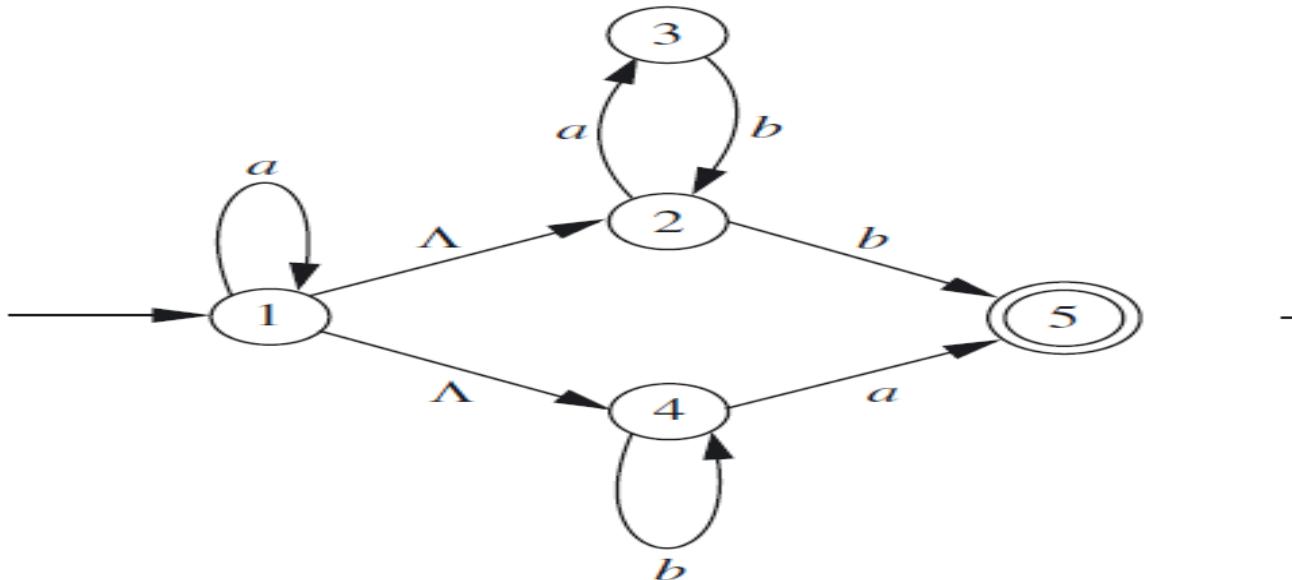


		a			b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$

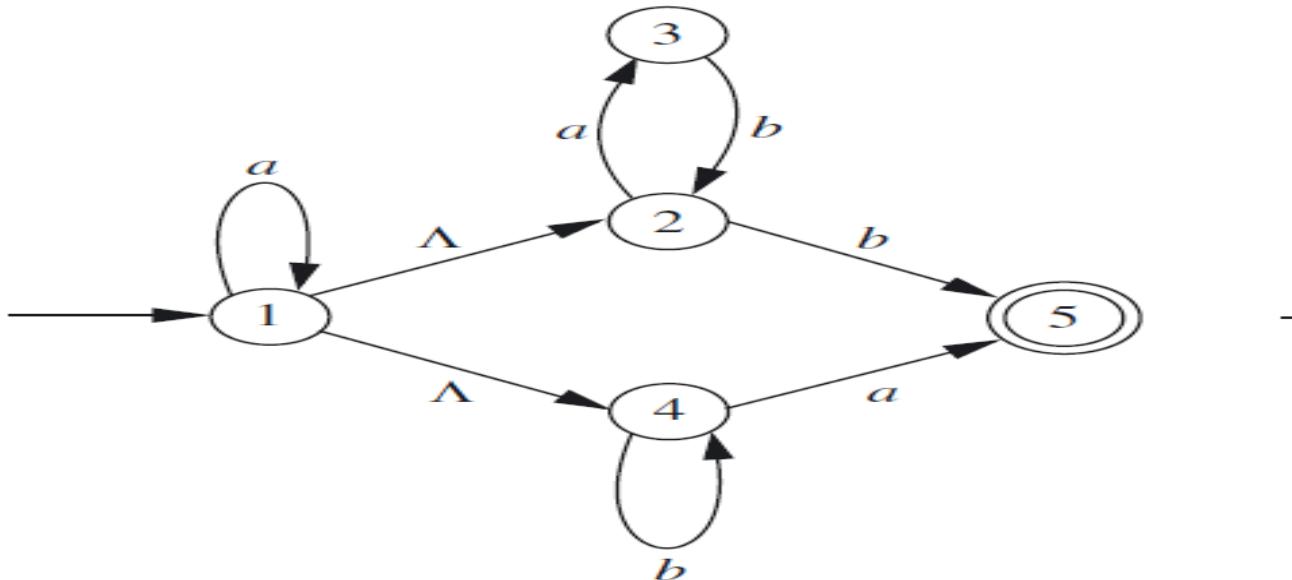




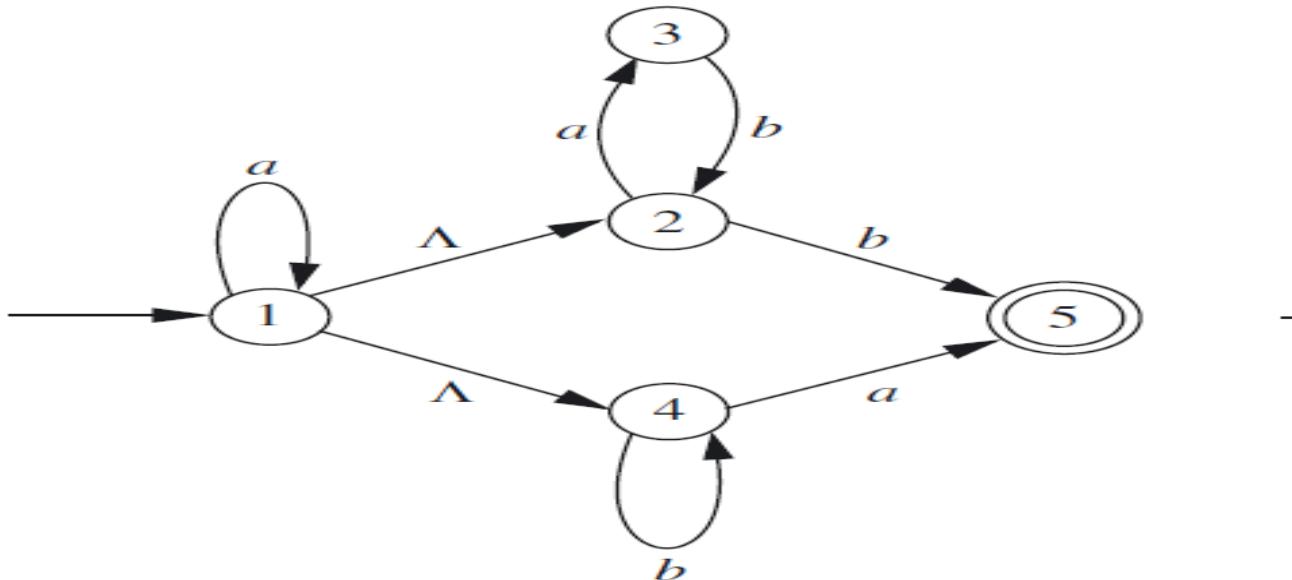




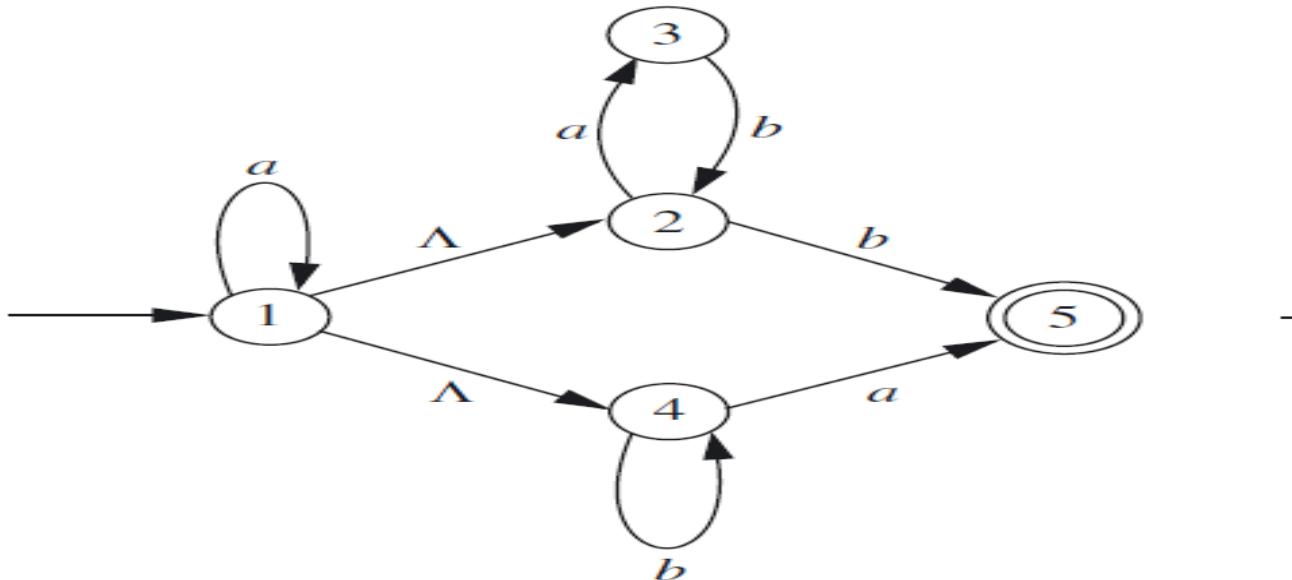




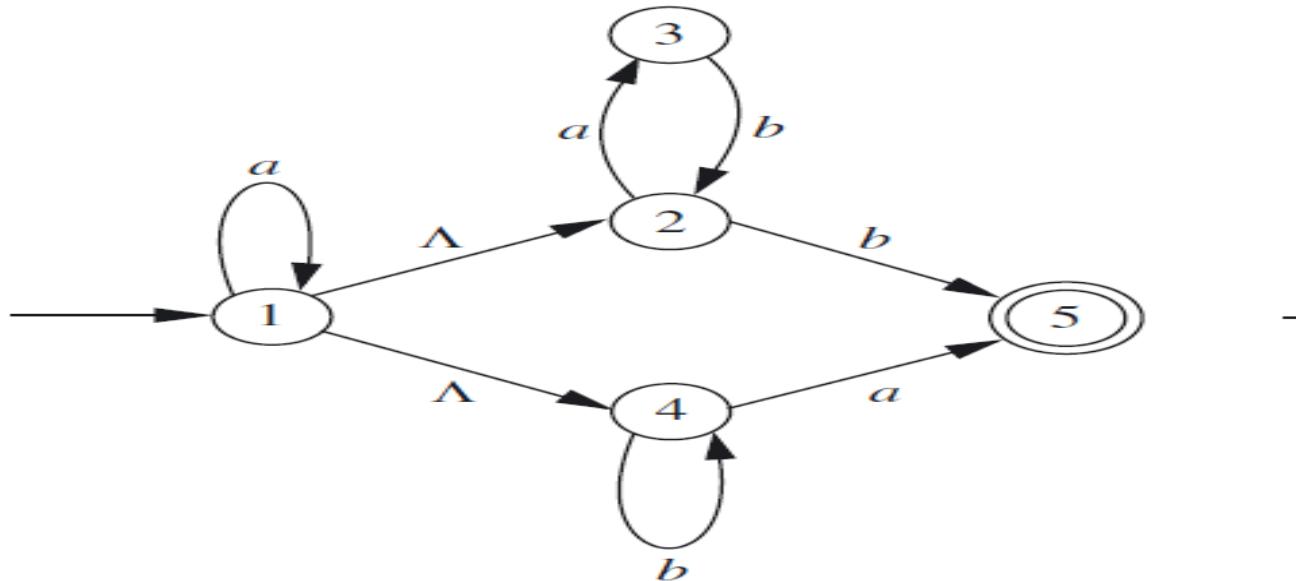
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$2,4,5$	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$



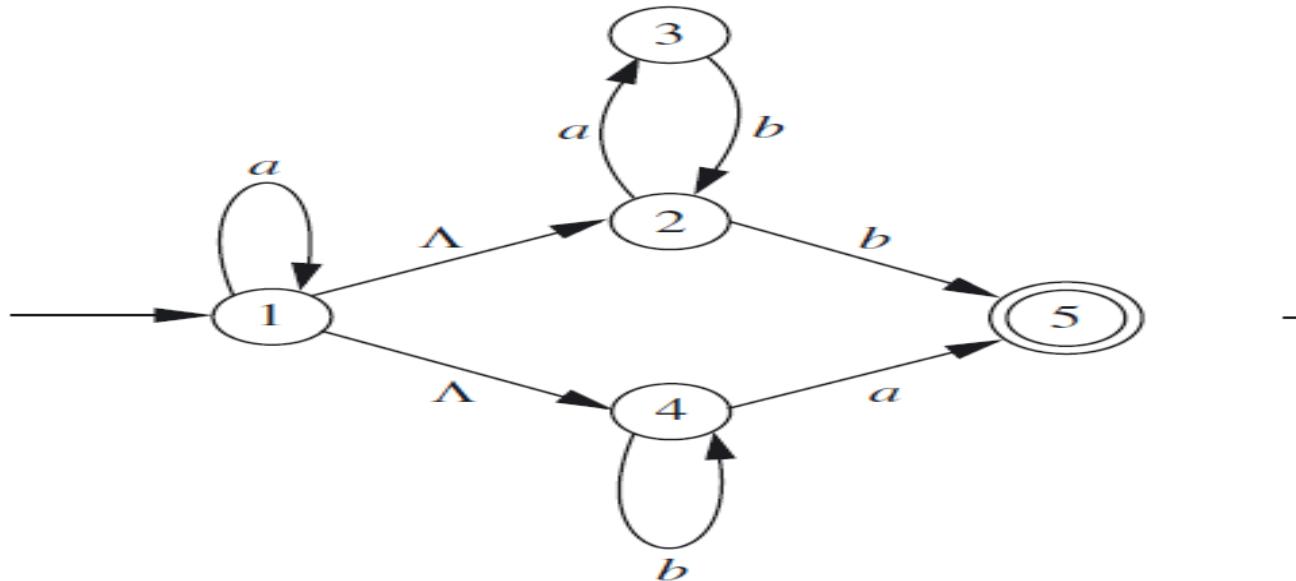
		a			b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$2,4,5$	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$



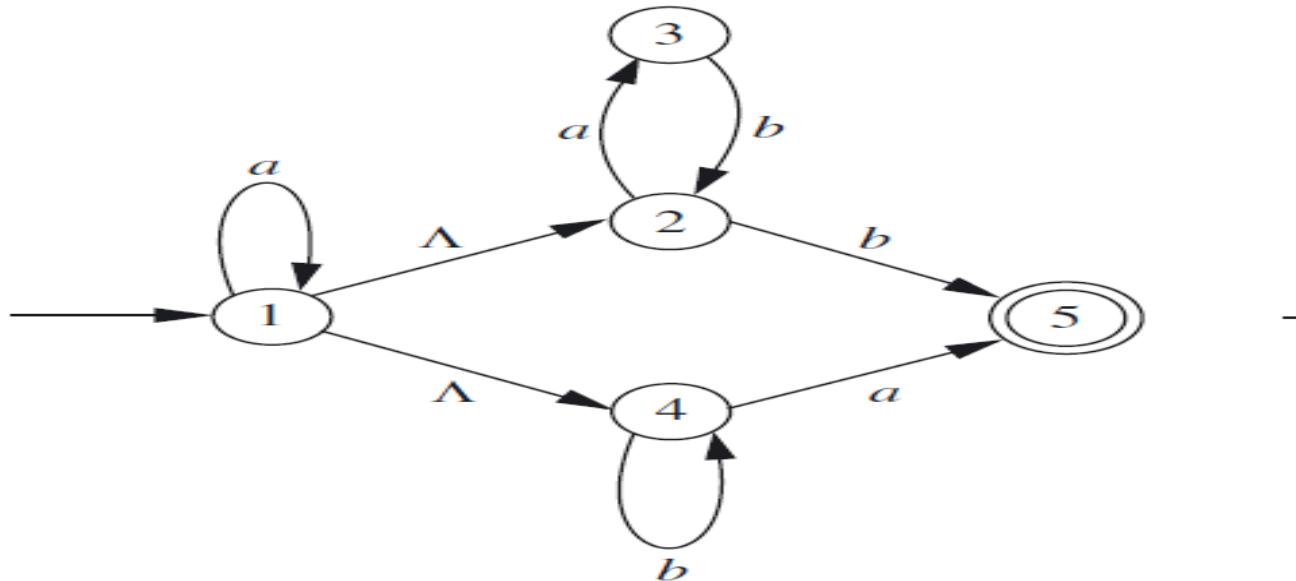
		a			b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$2,4,5$	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$						



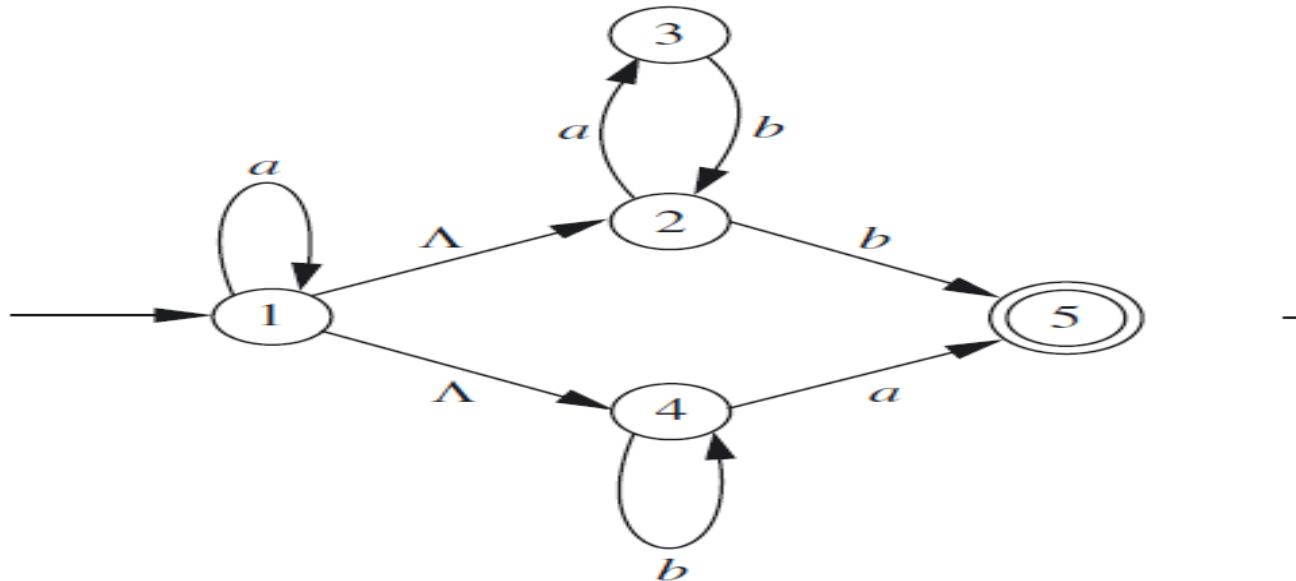
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$					



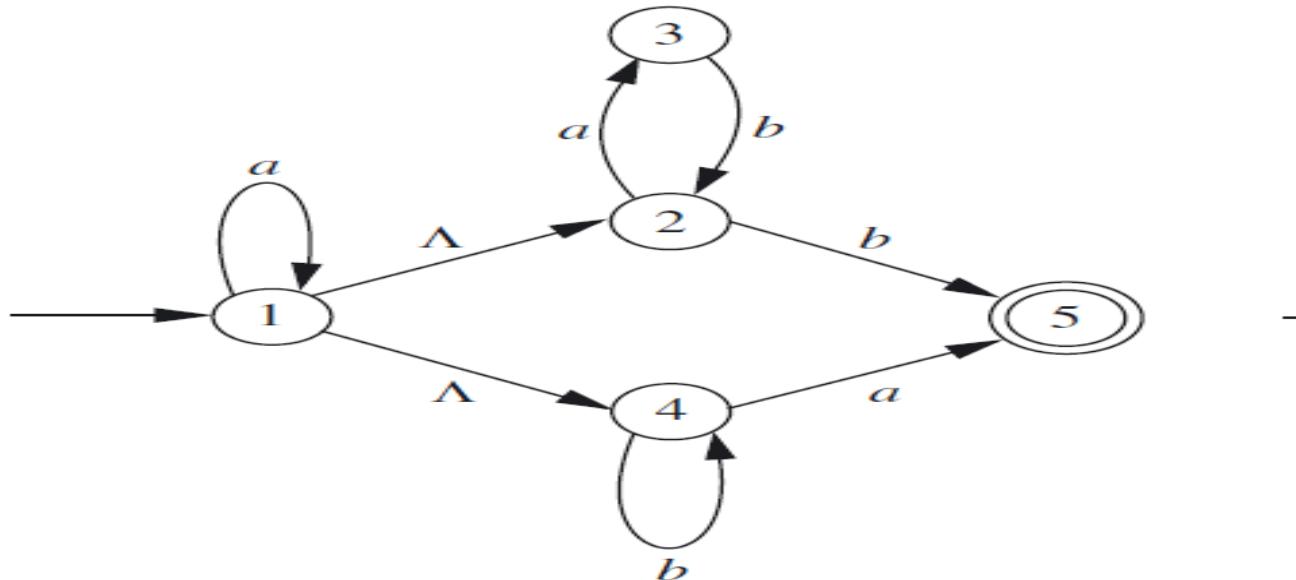
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$				



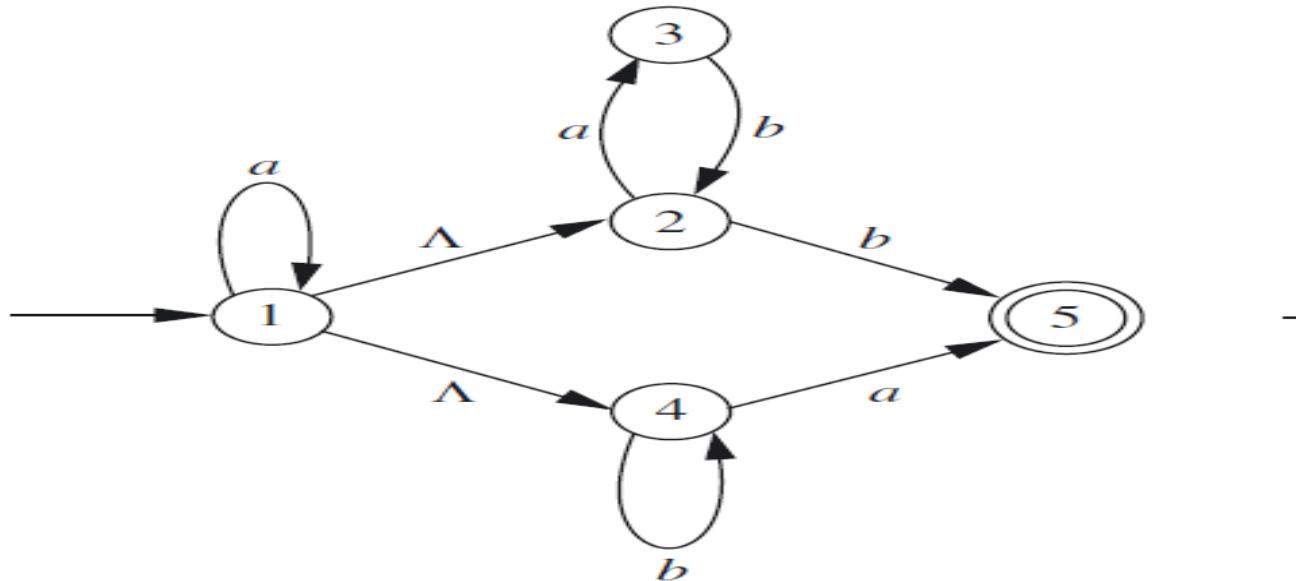
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$			



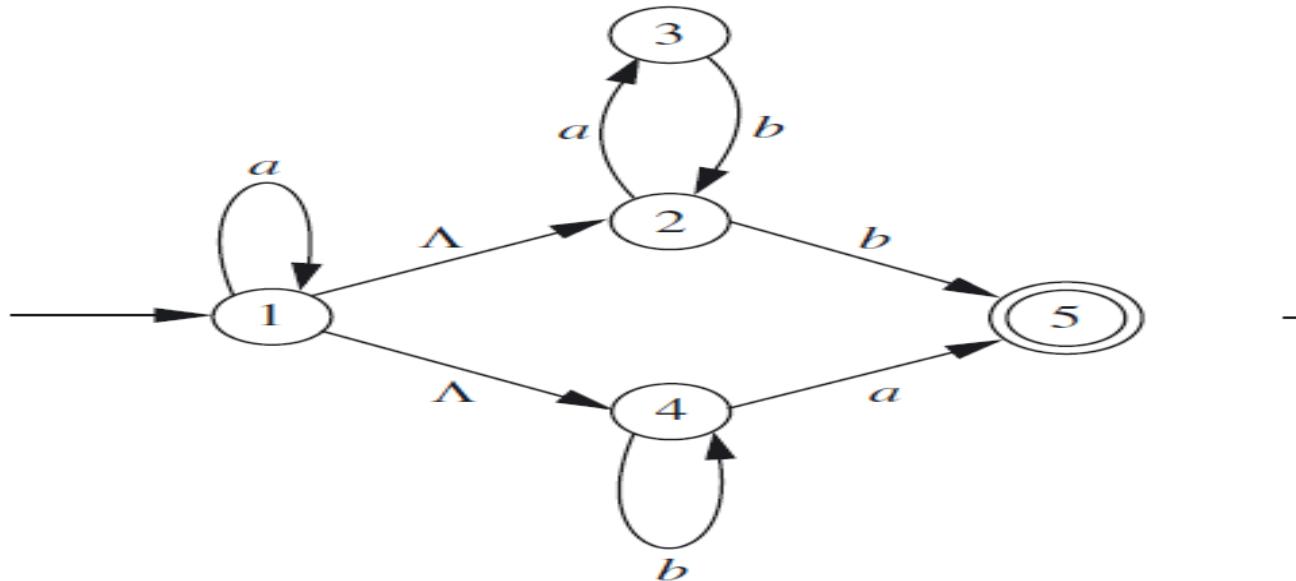
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	



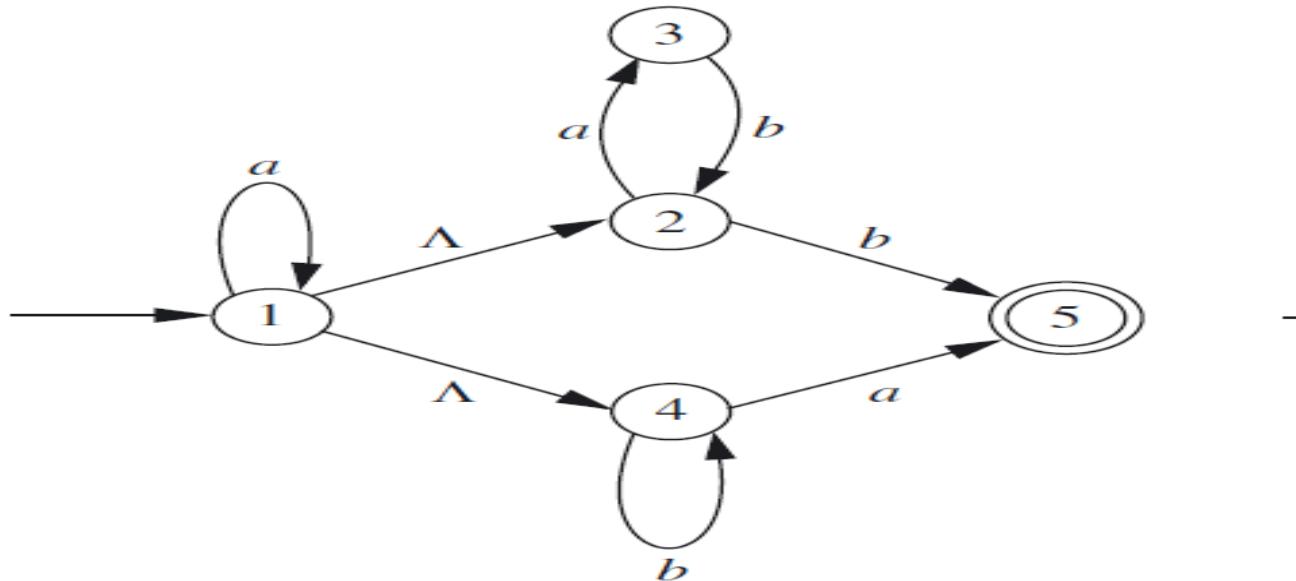
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$



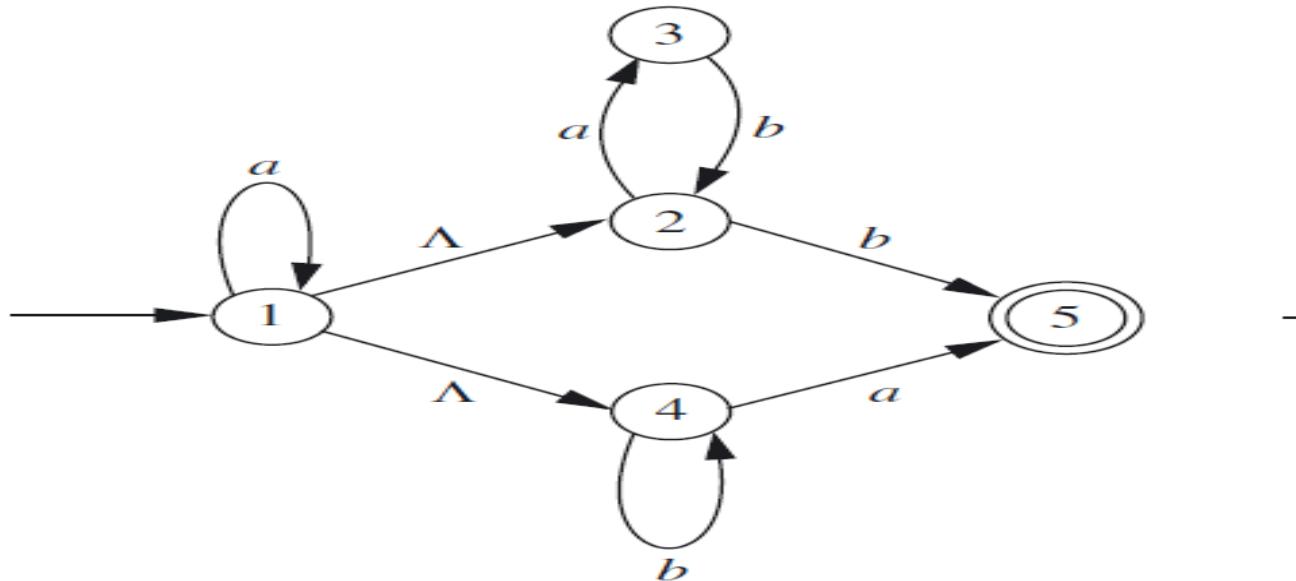
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$



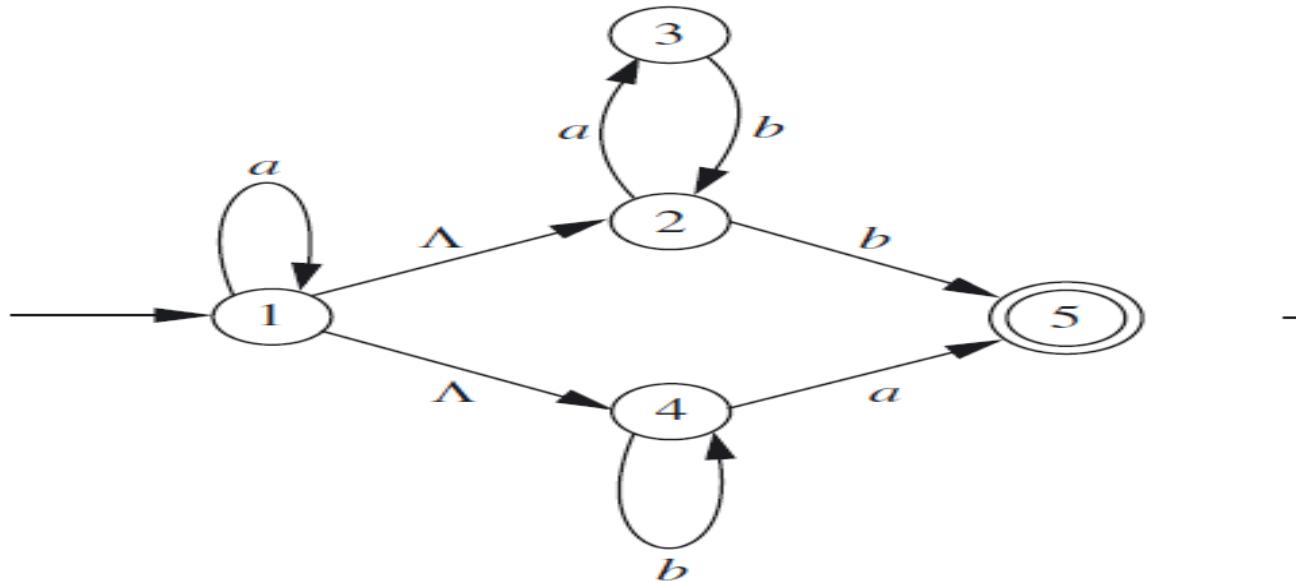
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4						



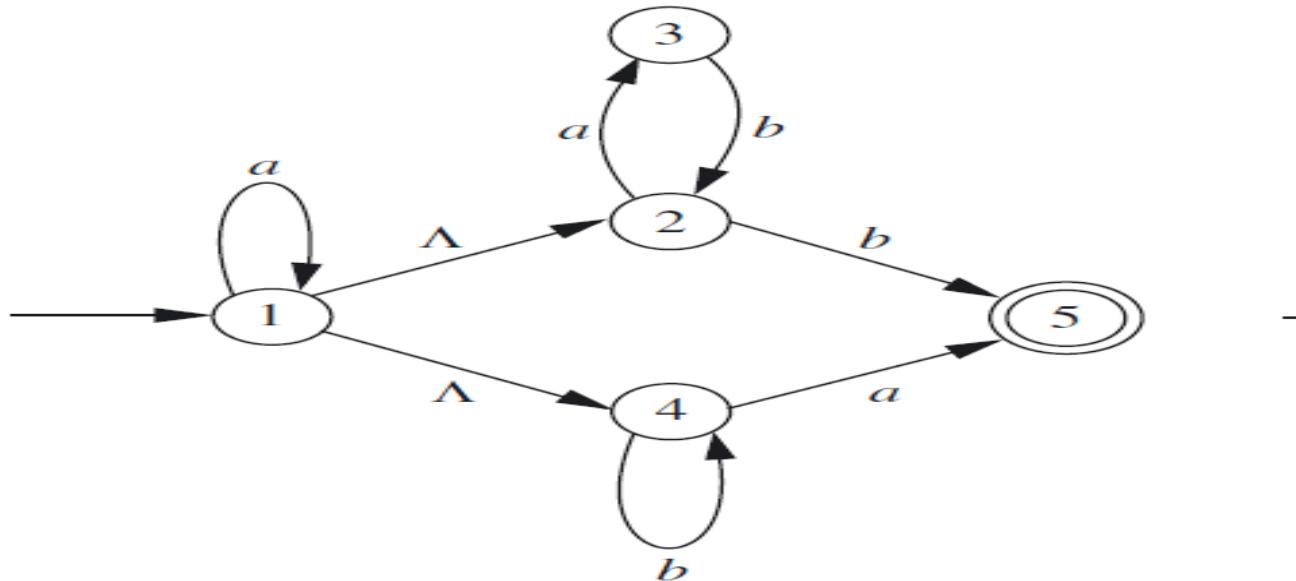
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$					



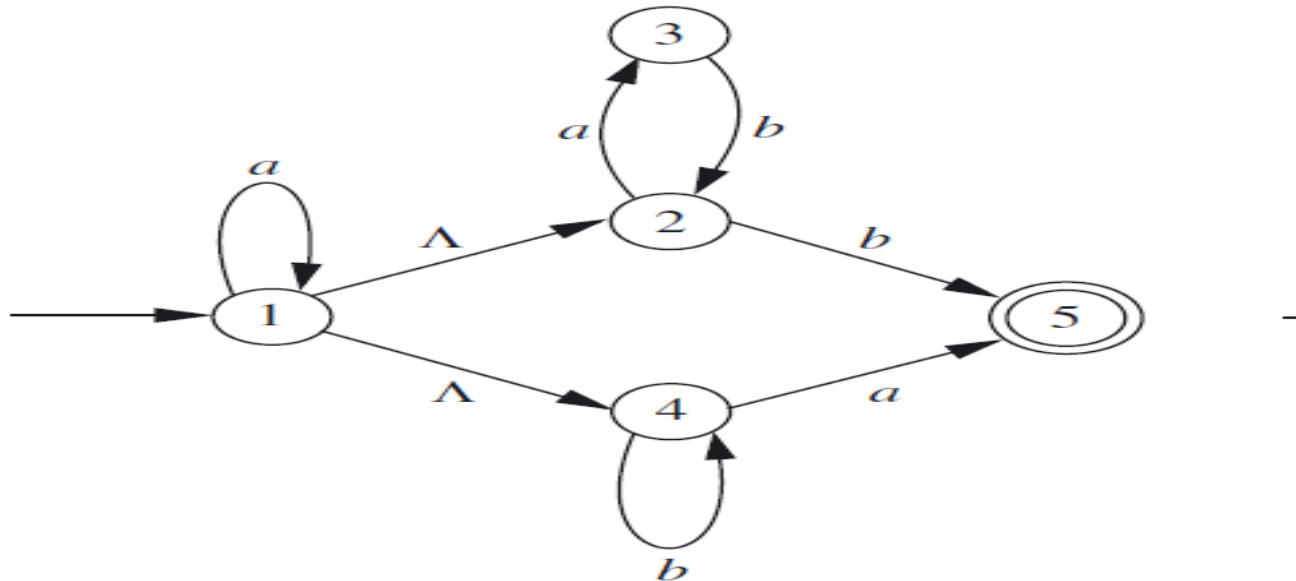
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$				



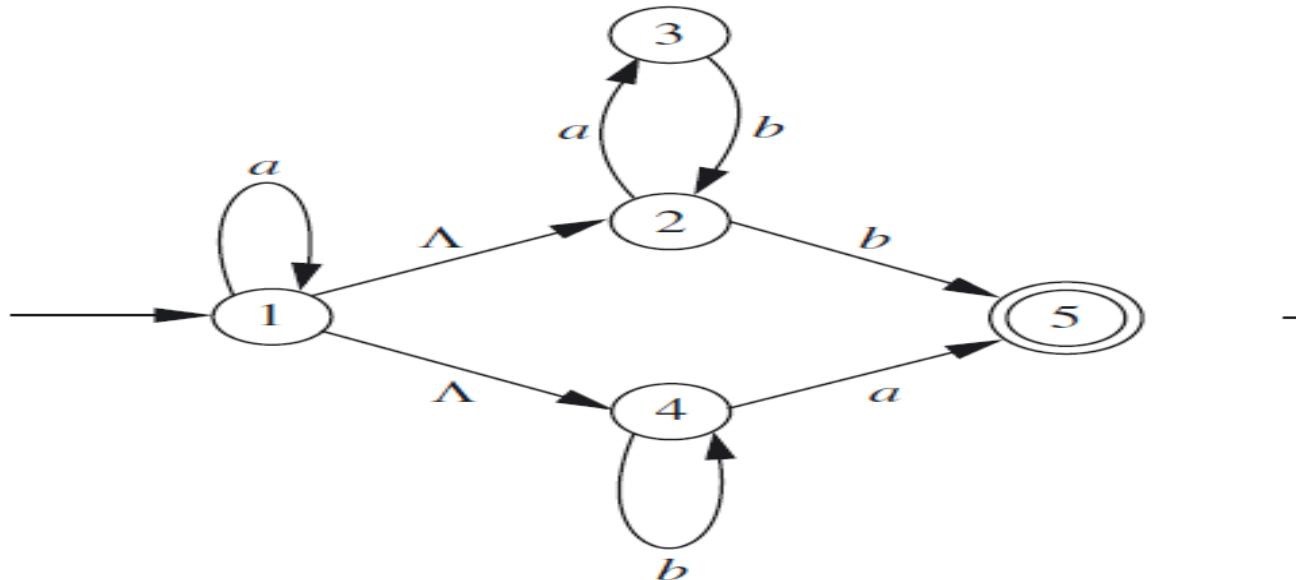
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$			



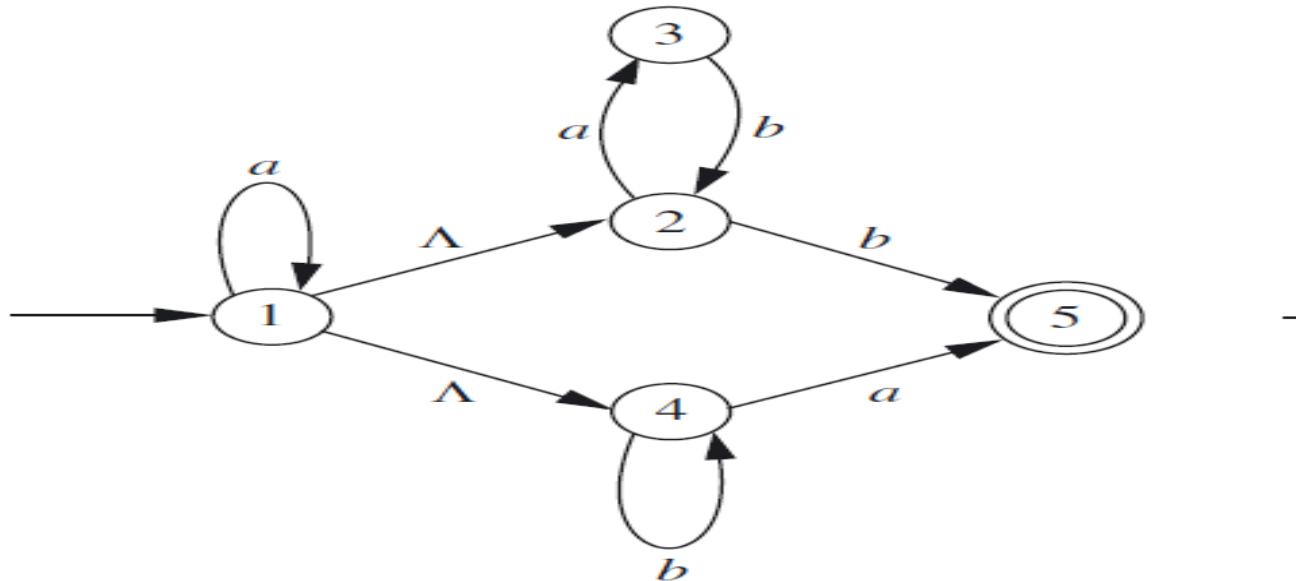
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	



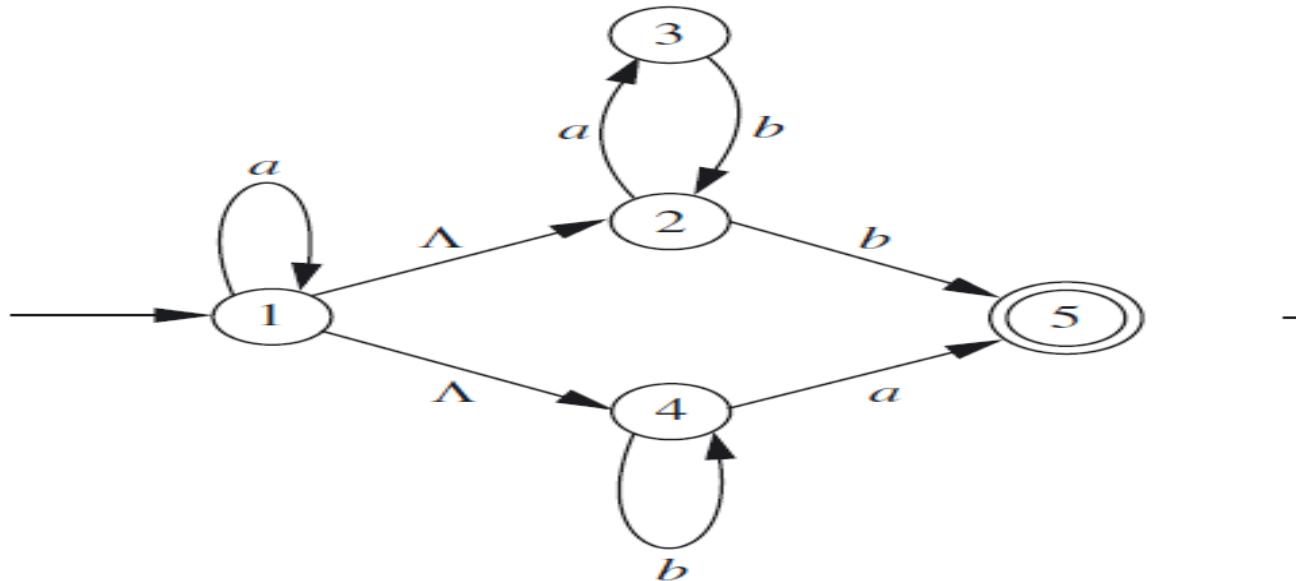
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$



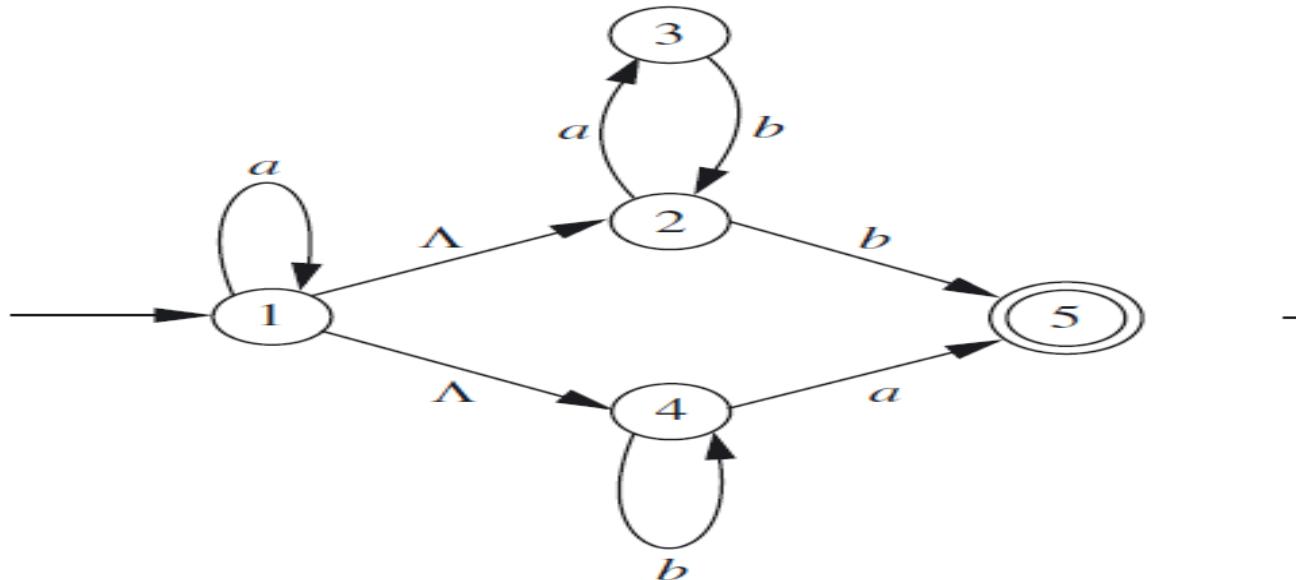
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$



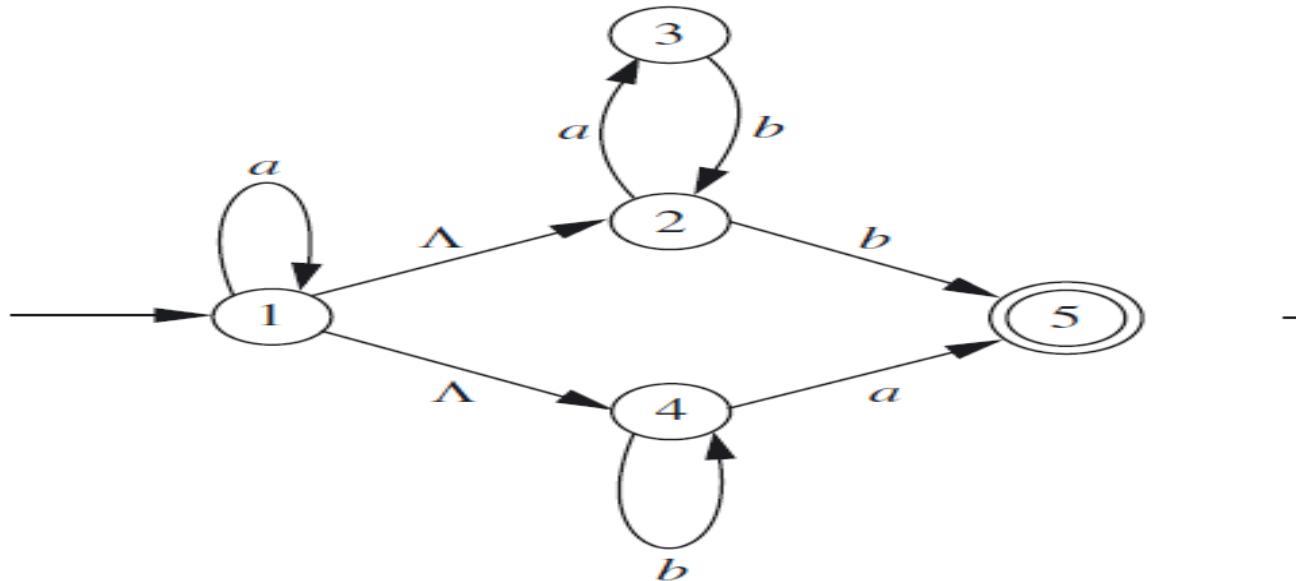
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5						



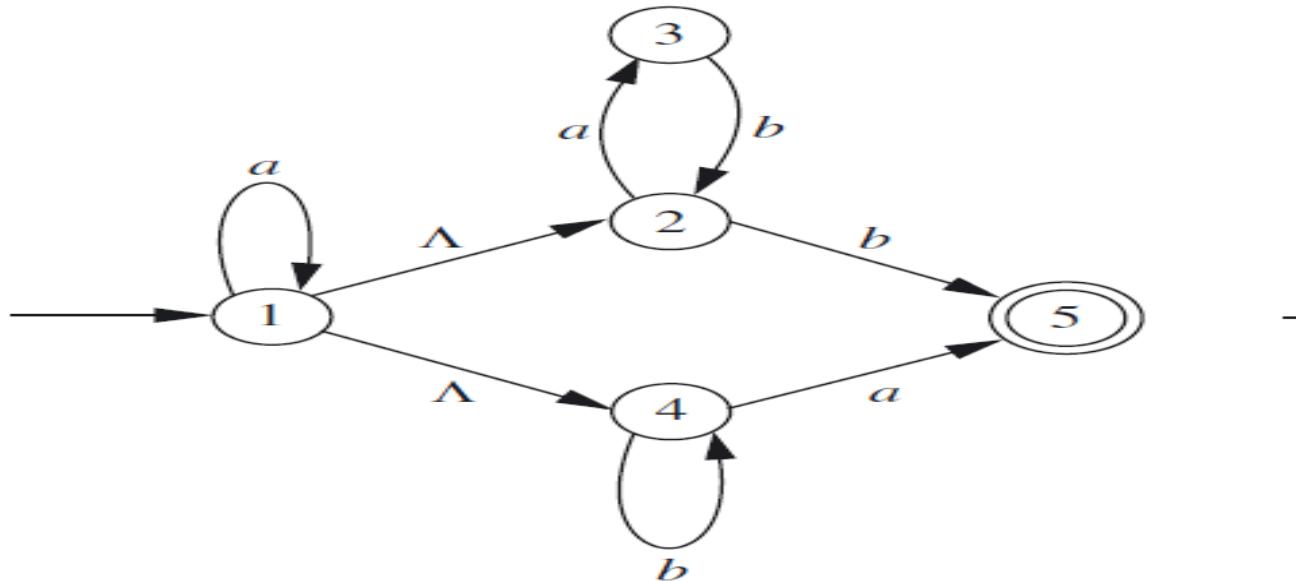
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$					



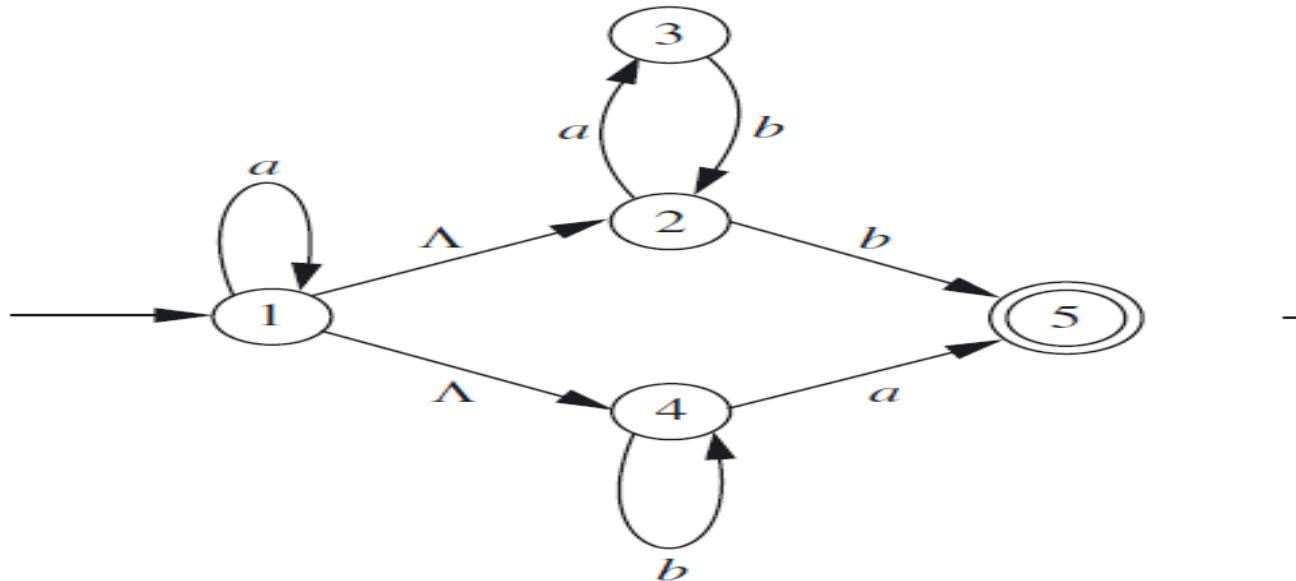
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$			



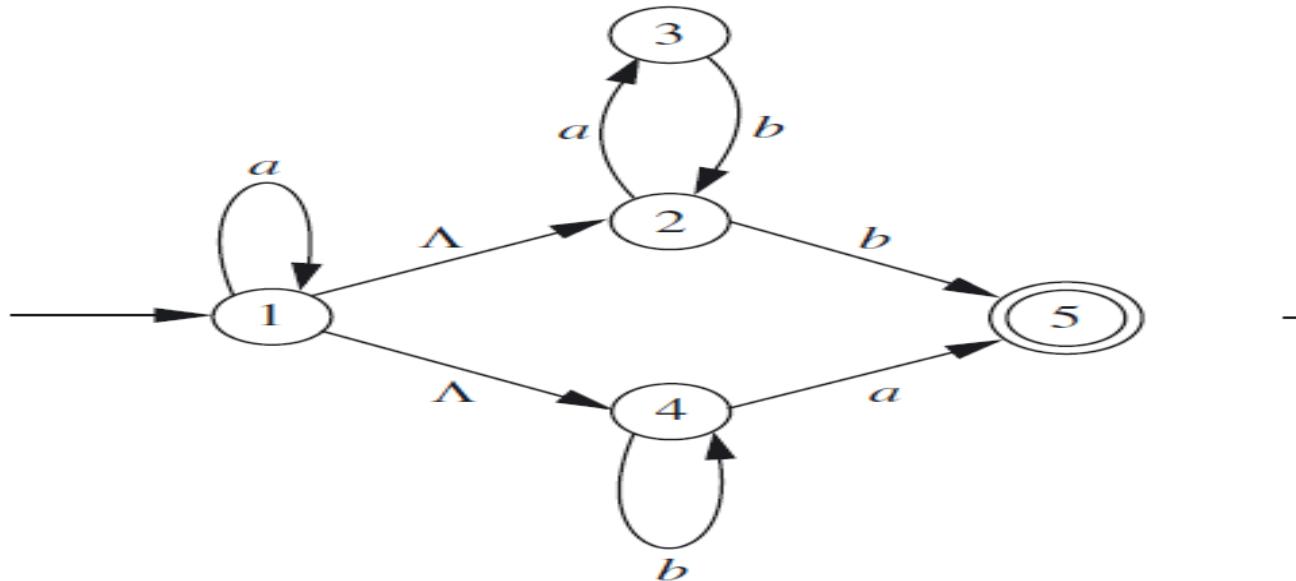
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$



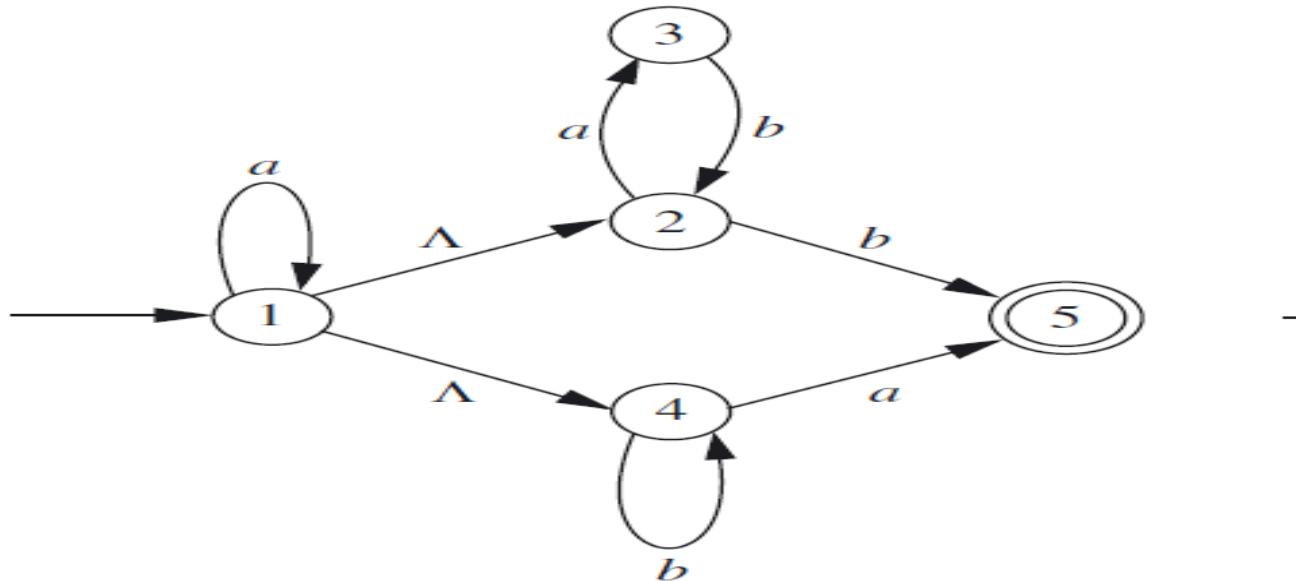
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$



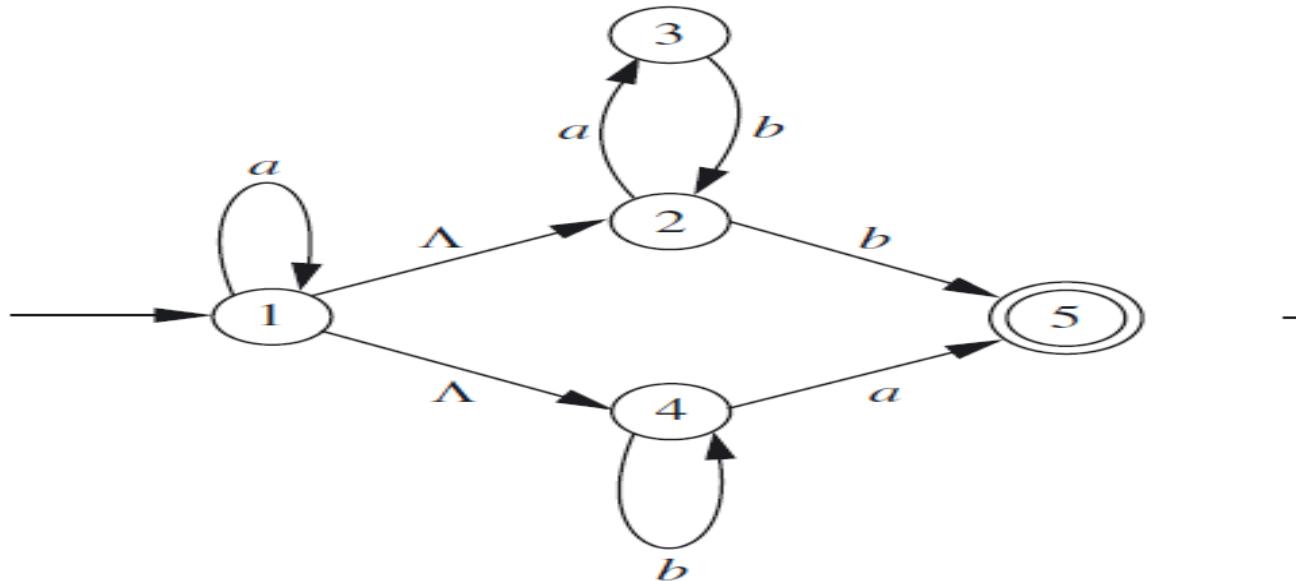
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$						



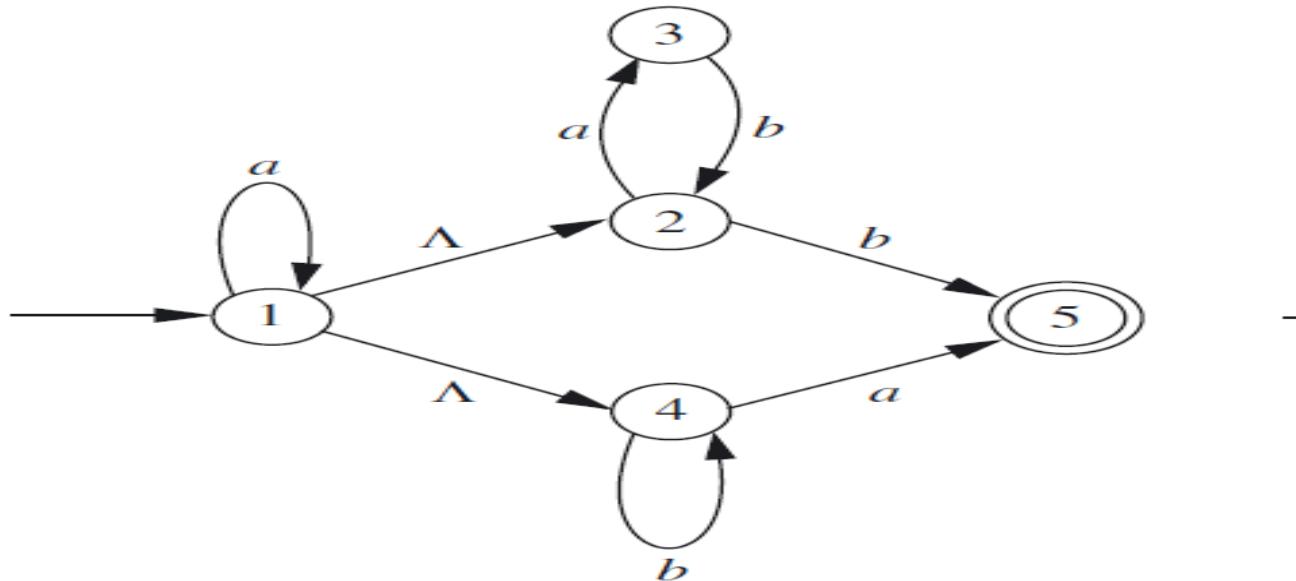
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$					



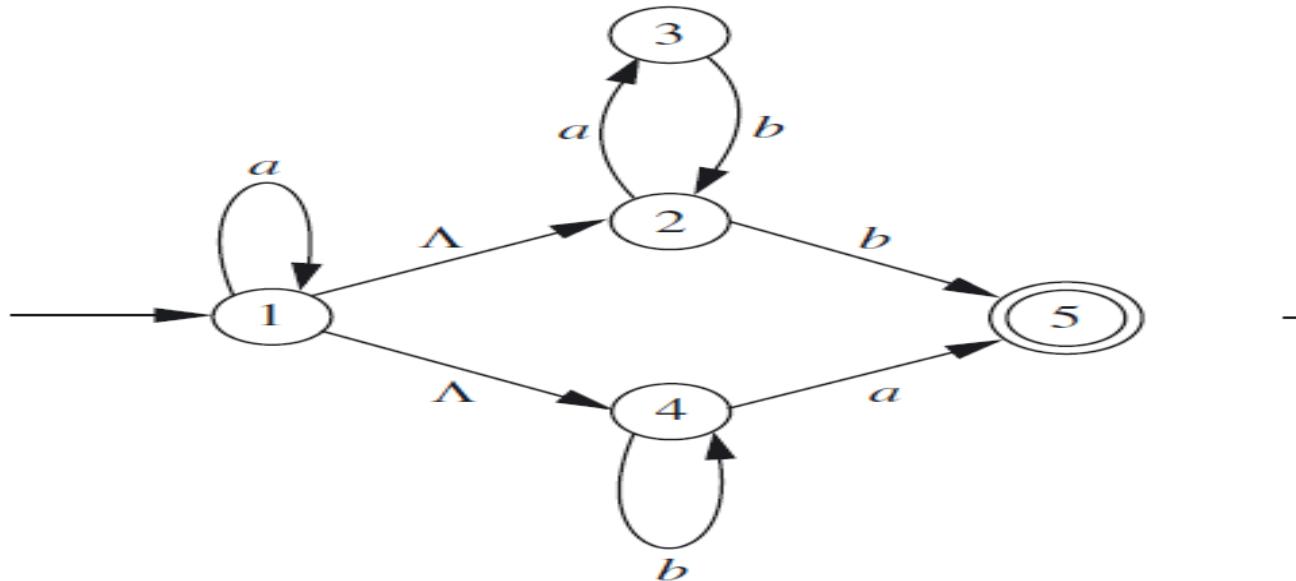
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$			



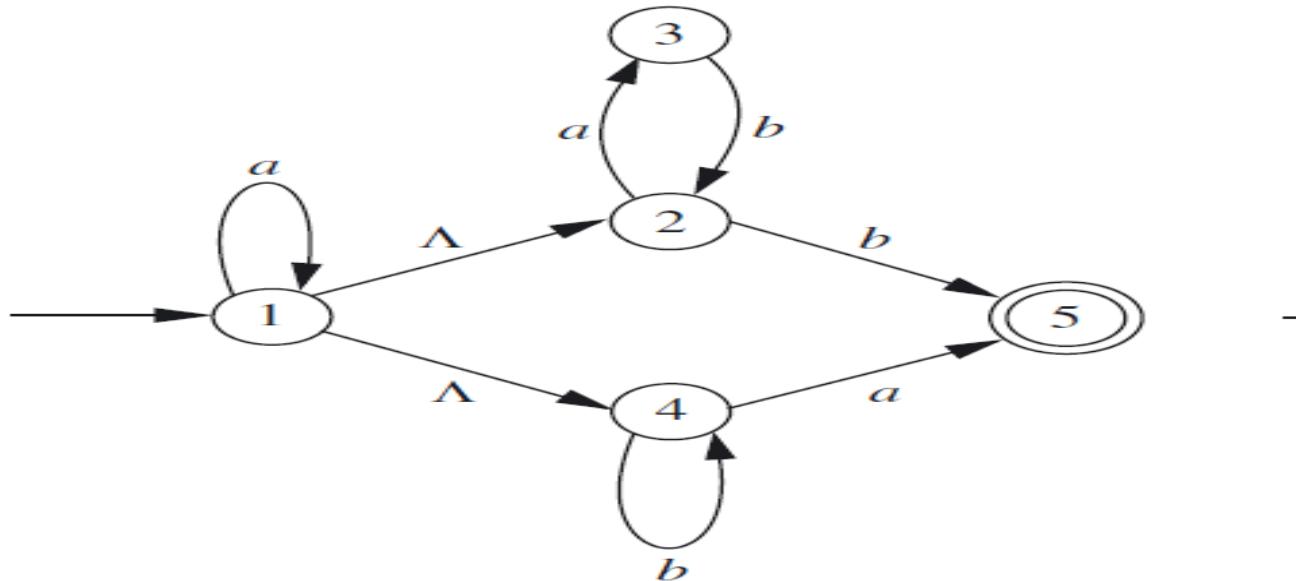
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	



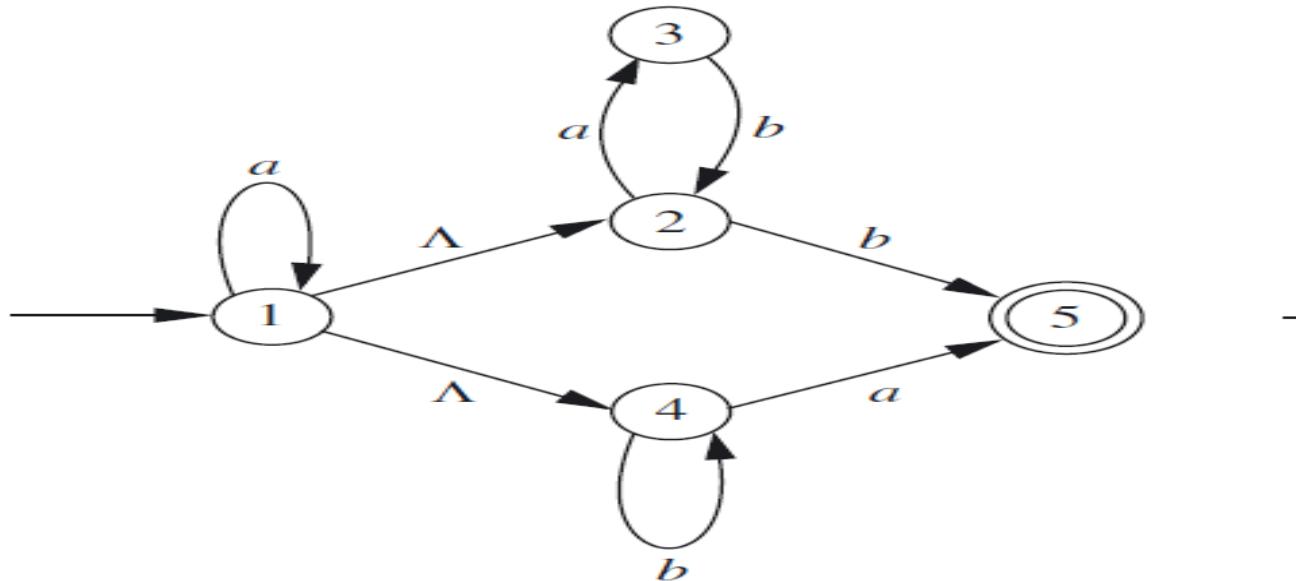
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$



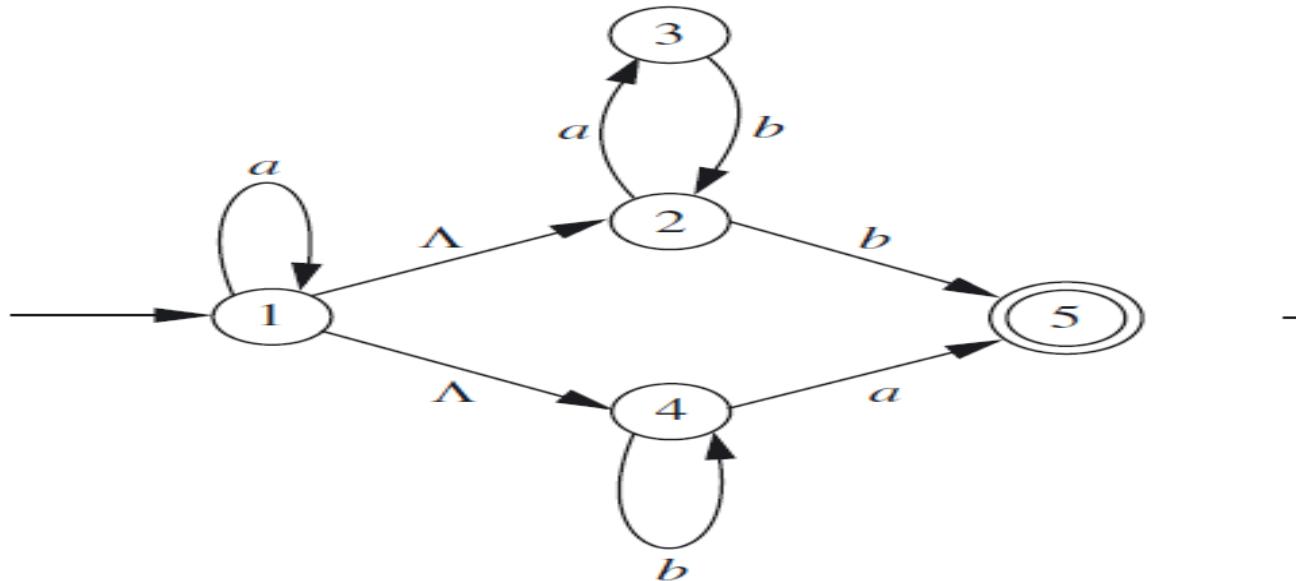
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$



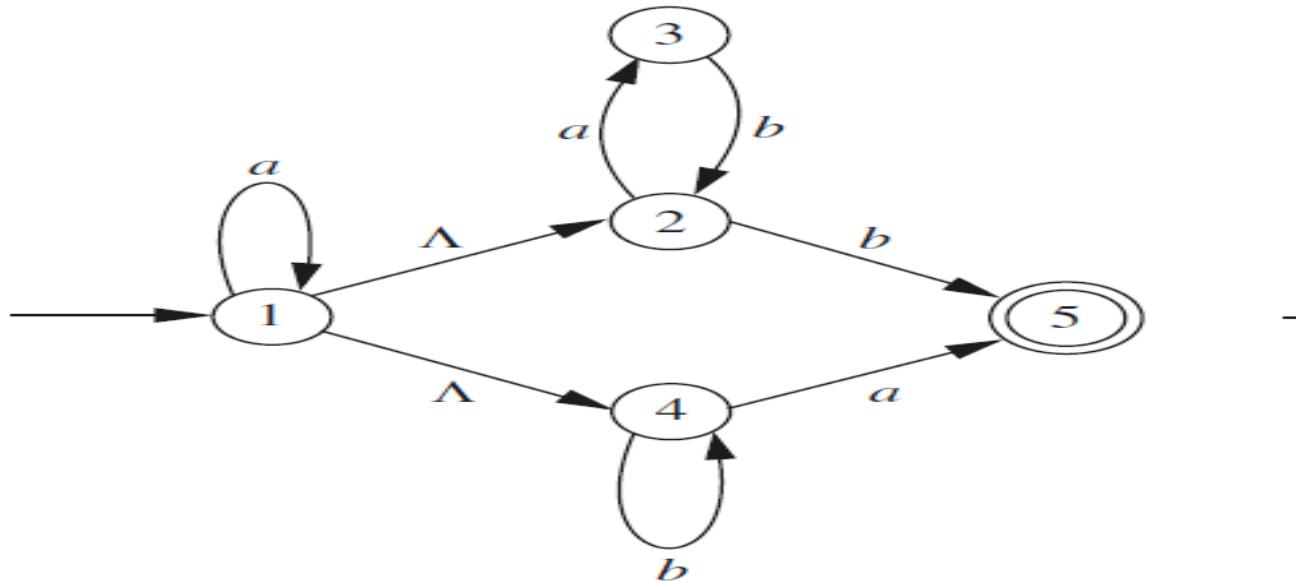
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2						



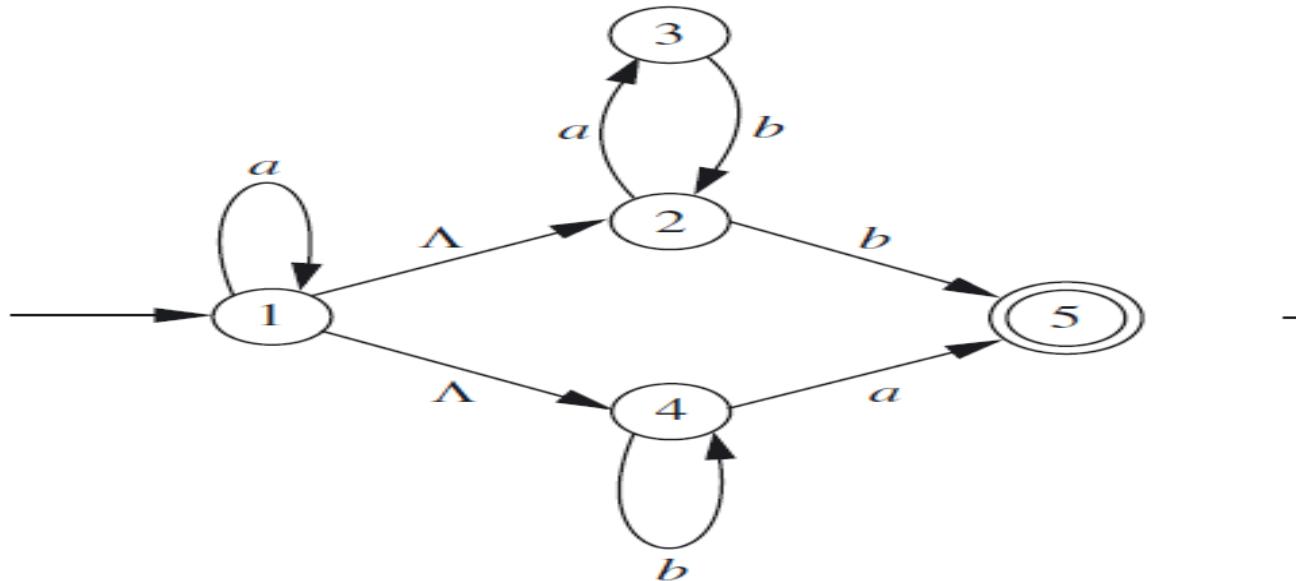
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$					



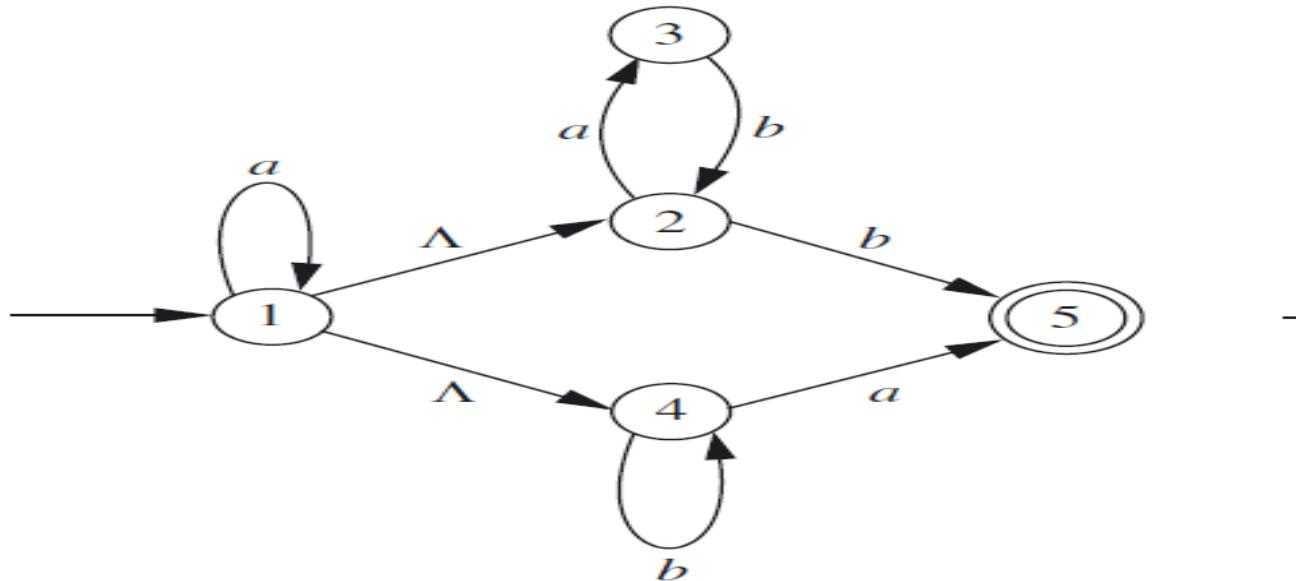
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$				



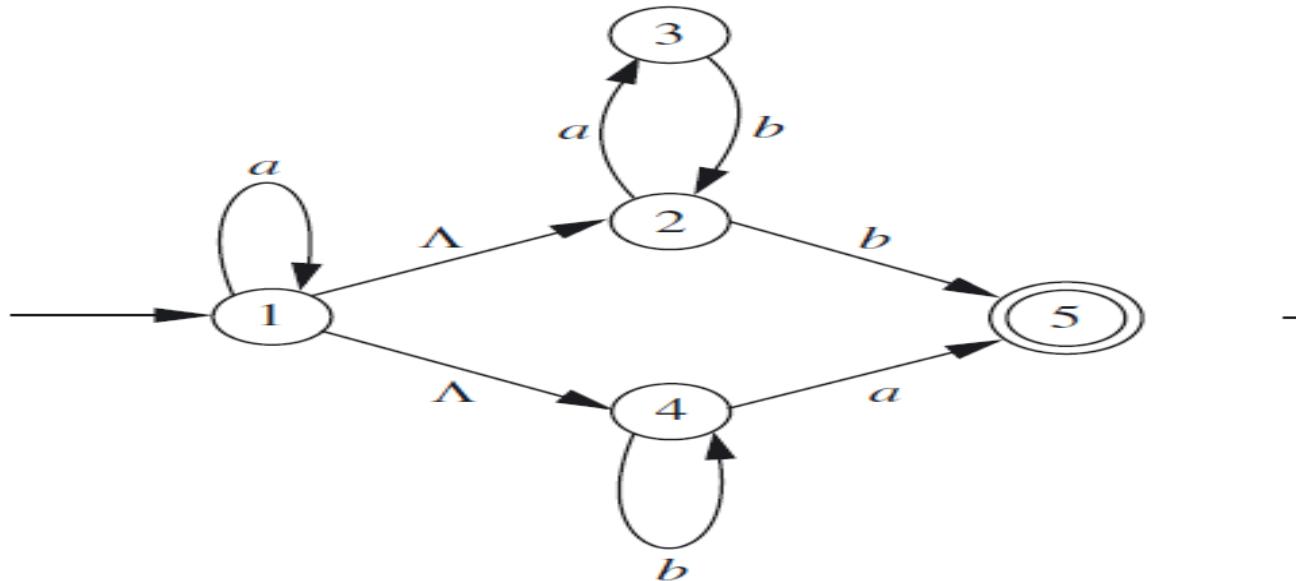
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$			



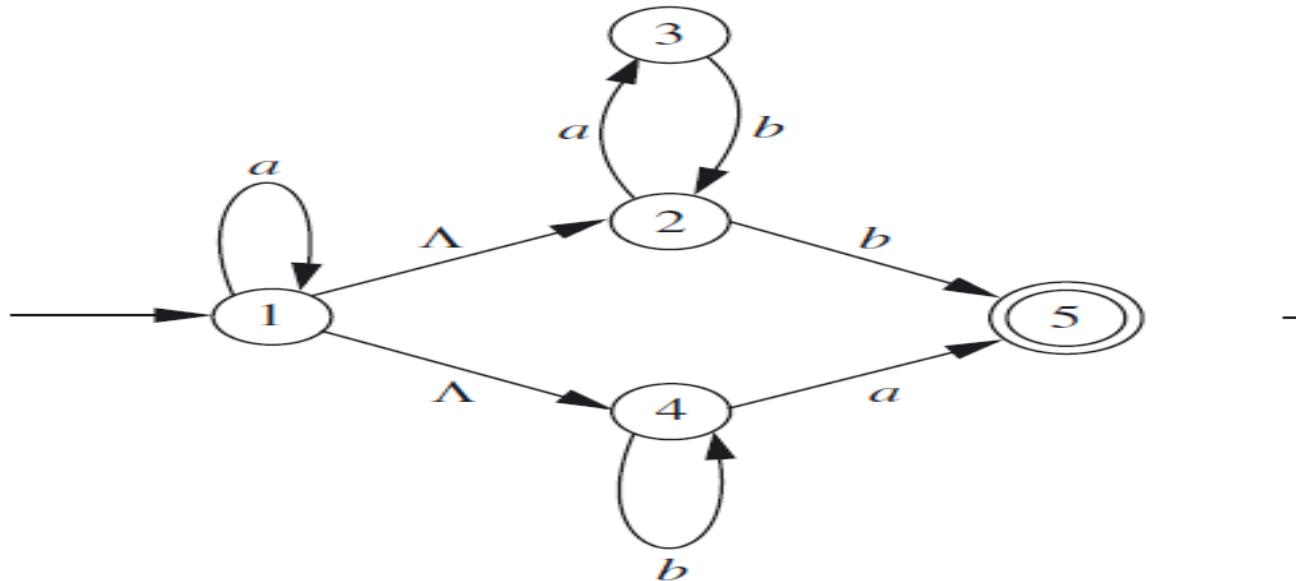
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$		



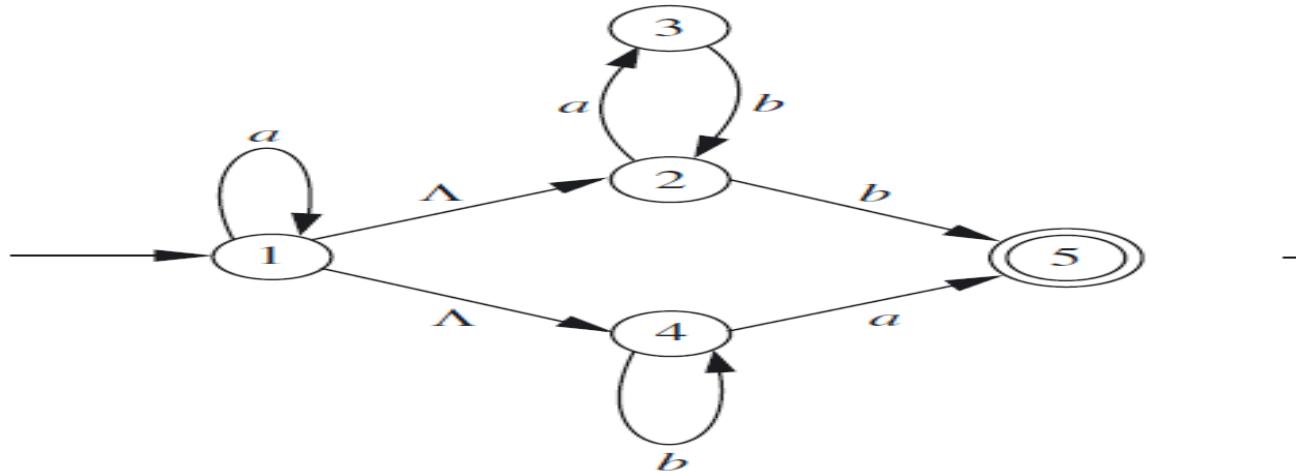
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	



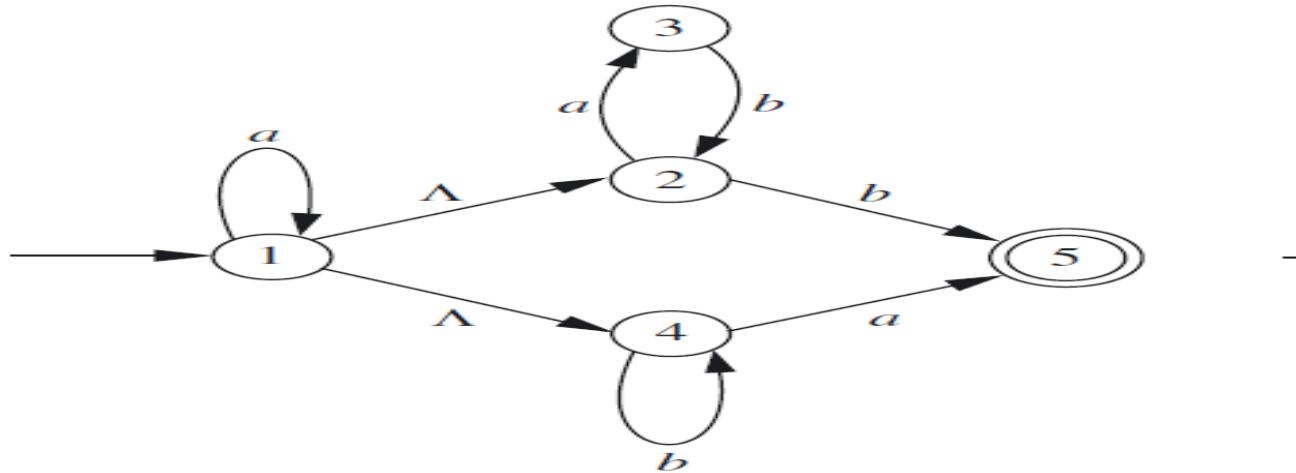
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$



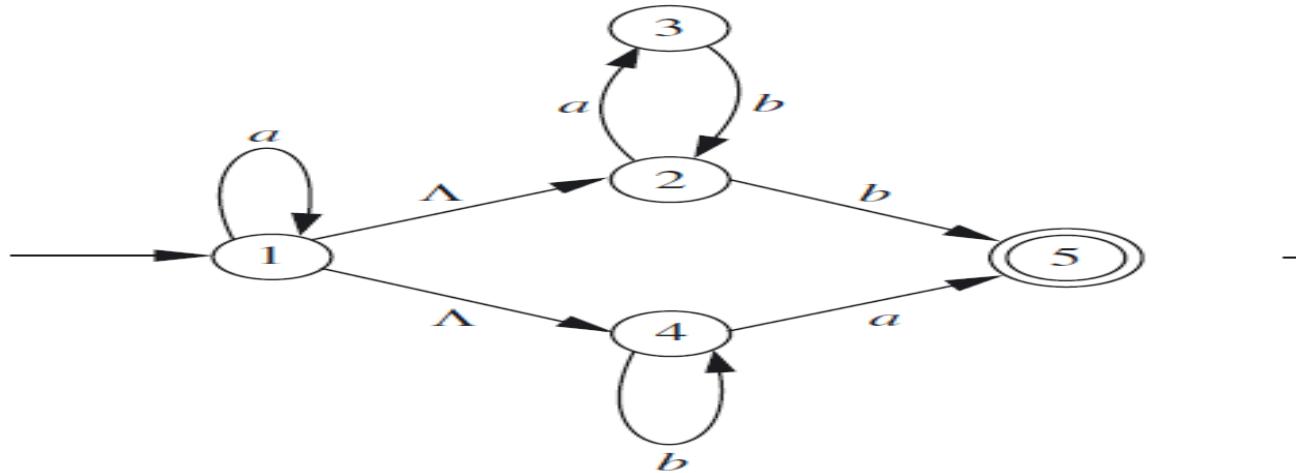
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$



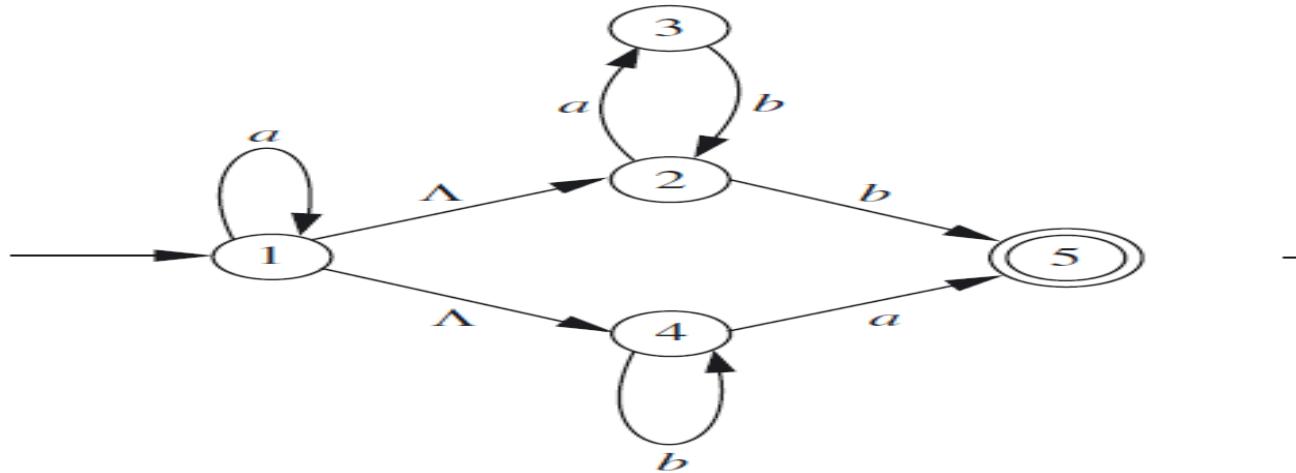
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
$\{3\}$						



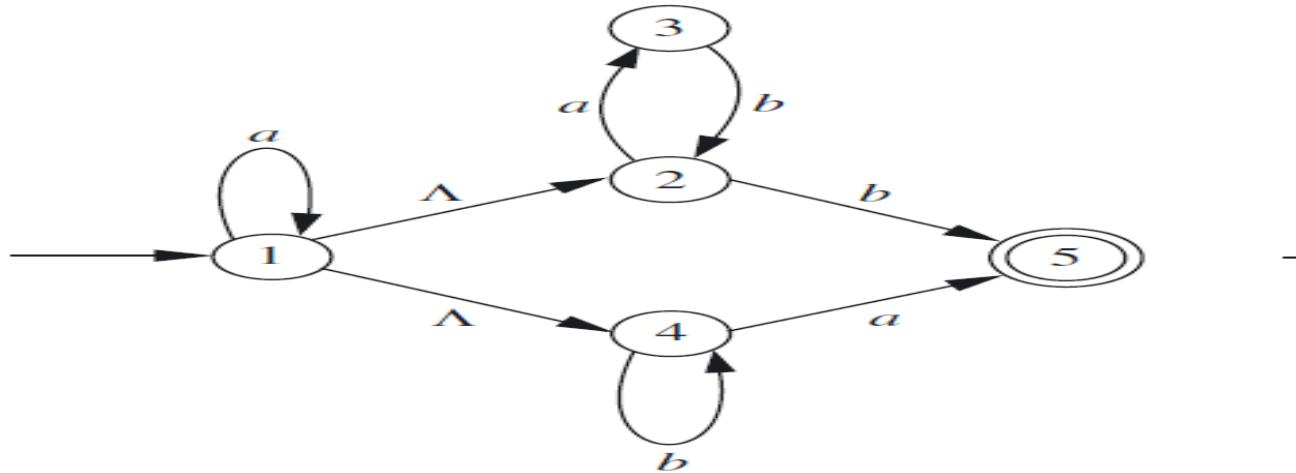
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
$\{3\}$	$\{3\}$					



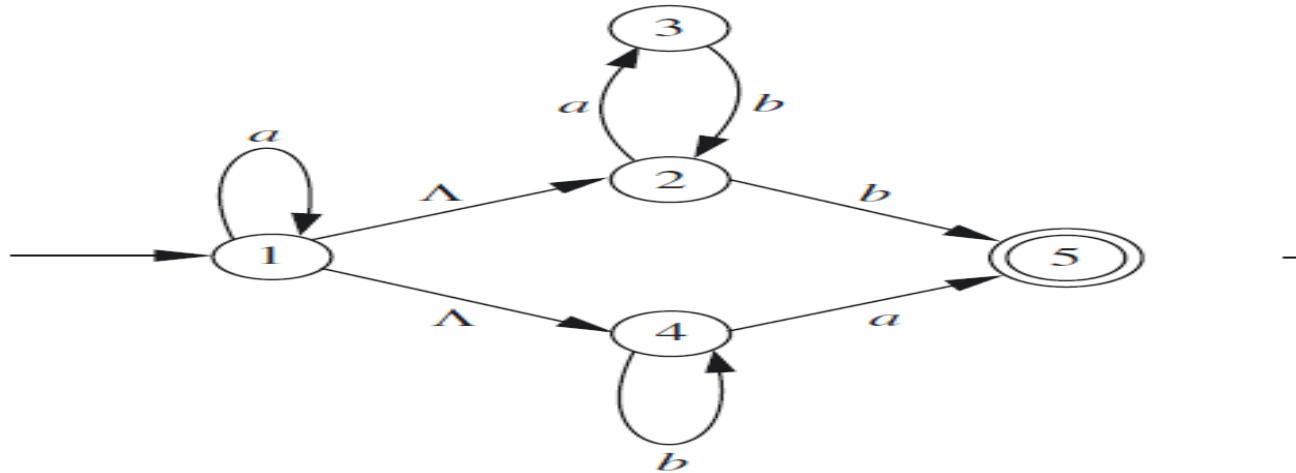
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
$\{3\}$	$\{3\}$	$\Phi$				



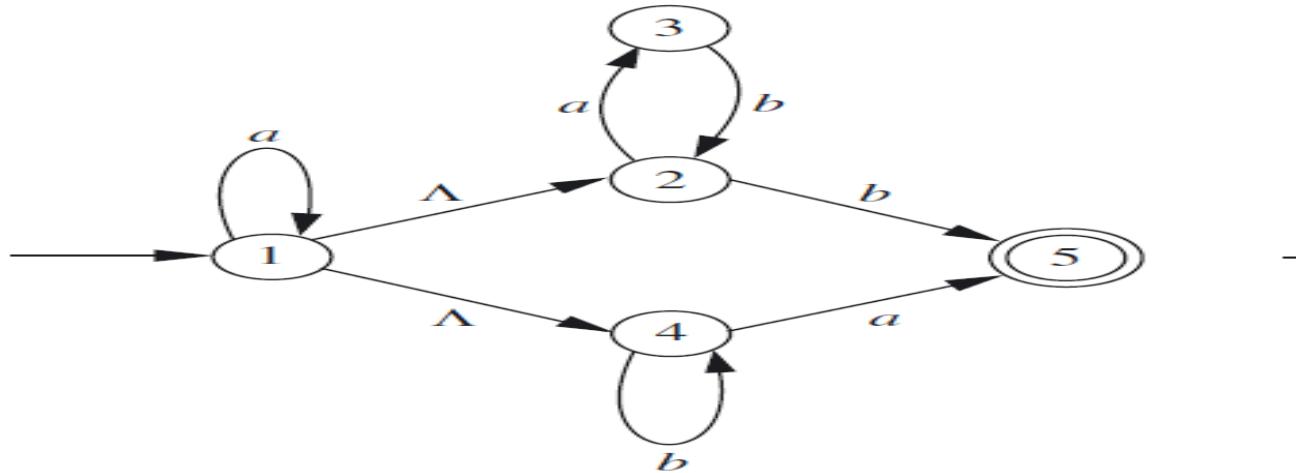
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
$\{3\}$	$\{3\}$	$\Phi$	$\Phi$			



		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
$\{3\}$	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	



		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
$\{3\}$	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$

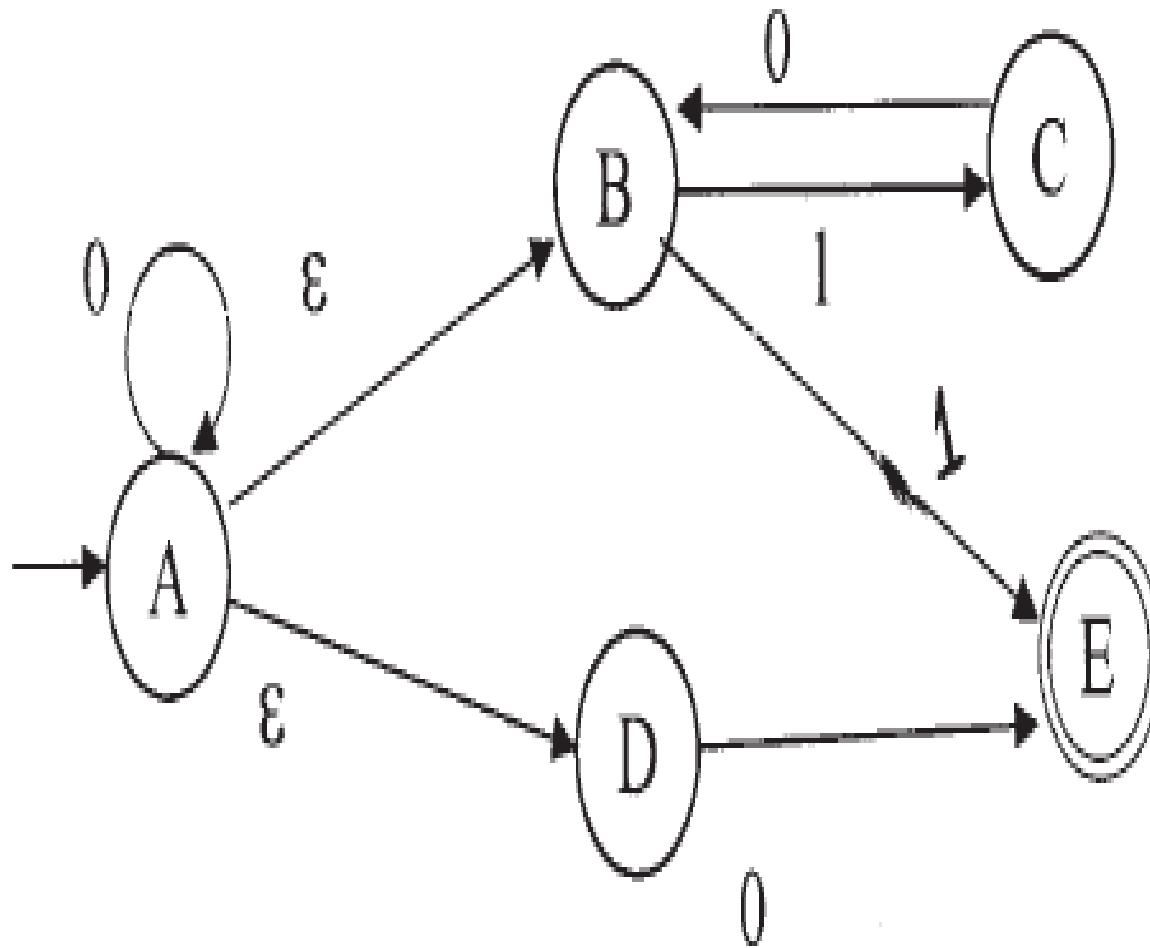


		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$2,4,5$	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
$\{3\}$	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$

02/07/2019Absent

- 3,16,17,27,33,57,62,71,74

Convert the given NFA- $\epsilon$  to an NFA.



# Review

- DFA, NFA and NFA- $\lambda$  Equivalency
- Conversion of NFA to DFA, NFA- $\lambda$  to DFA

# THEORY OF COMPUTATION

---

Unit I

## FINITE STATE MACHINES

---

**Jagadish Kashinath Kamble** (M.Tech, IIT Kharagpur)

Dept. of Information Technology,  
Pune Institute of Computer Technology, Pune

## Finite Automata

Finite Automata with o/p

Moore  
Machine

Mealy  
Machine

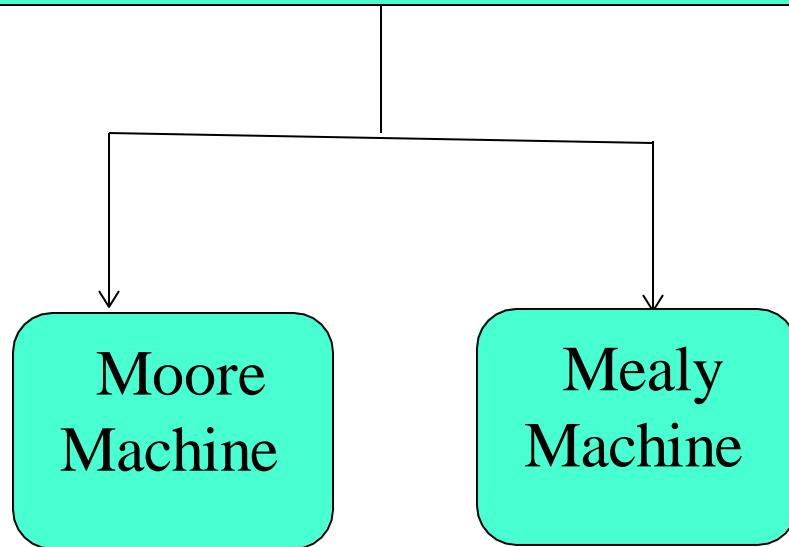
Finite Automata without o/p

Deterministic  
FA

Non Deterministic  
FA- $\epsilon$

Non-  
Deterministic  
FA

## Finite Automata with o/p



# • Moore Machine and Mealy machine

**Definition:-**

$$(Q, \Sigma, q_0, \delta, \Delta, \gamma)$$

Where,

$Q$ =set of states,  $\Sigma$ =i/p alphabet,  $q_0$ = start/initial state,

$\delta=Q \times \Sigma \rightarrow Q$  (transition function),

$\Delta$ =o/p alphabet,  $\gamma$  =o/p function

• Difference between Mealy and Moore machine is  $\gamma$  (o/p function)

$\gamma$  (o/p function) for Moore machine is  $\gamma : Q \rightarrow \Delta$

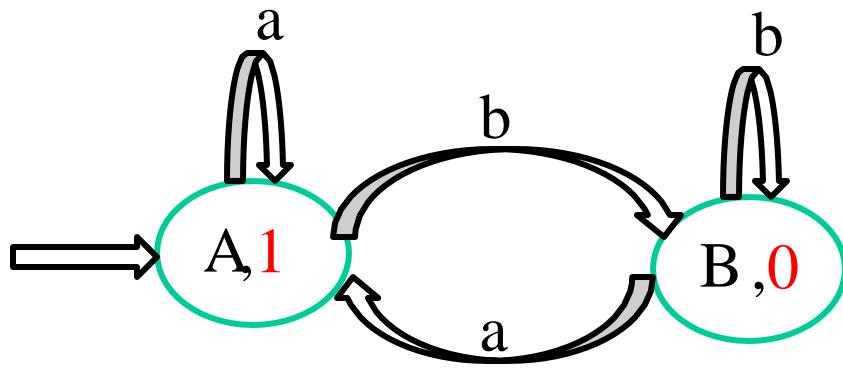
(outputs depend on only the present state)

$\gamma$ (o/p function) for Mealy machine is  $\gamma : Q \times \Sigma \rightarrow \Delta$

(Output depends on the present state as well as the present input)

• Moore and Mealy Machines are equally powerful

- Moore Machine



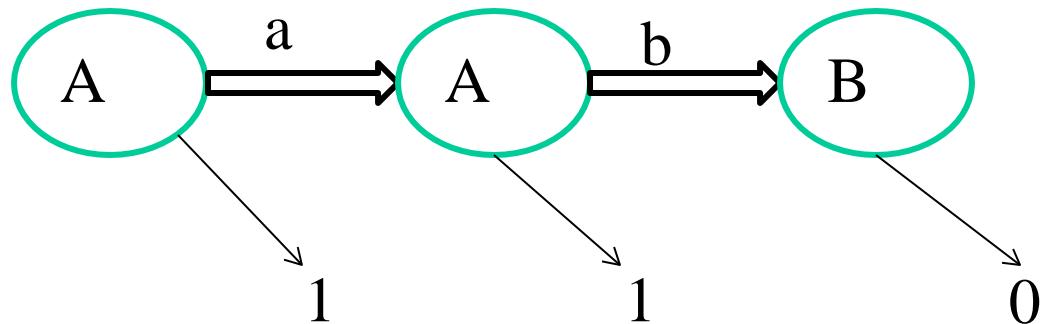
	Input		Output
	a	b	
A	A	B	1
B	A	B	0

$$\lambda: Q \rightarrow \Delta$$

$A \rightarrow 1$

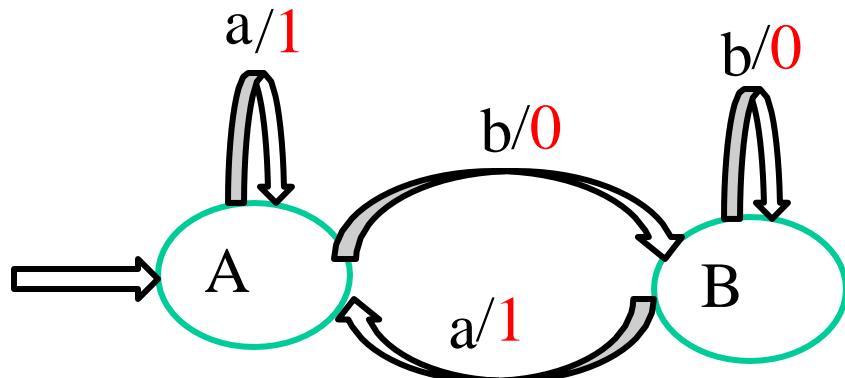
$B \rightarrow 0$

- string 'ab' on machine



- Applying string 'ab' on machine (as input), got the output as 110.
- If number of input symbol is N then Number of output symbol is N+1.

## • Mealy Machine



	Input	Output	Input	Output
	a		b	
A	A	1	B	0
B	A	1	B	0

$$\lambda: Q \times \Sigma \rightarrow \Delta$$

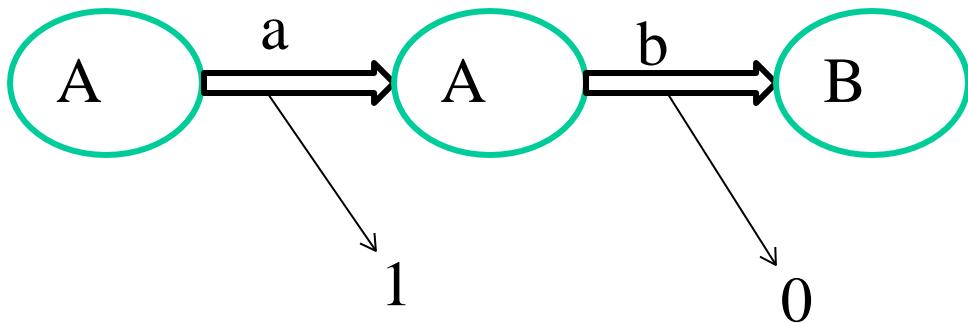
$$A, a \rightarrow 1$$

$$A, b \rightarrow 0$$

$$B, a \rightarrow 1$$

$$B, b \rightarrow 0$$

## • string 'ab' on machine

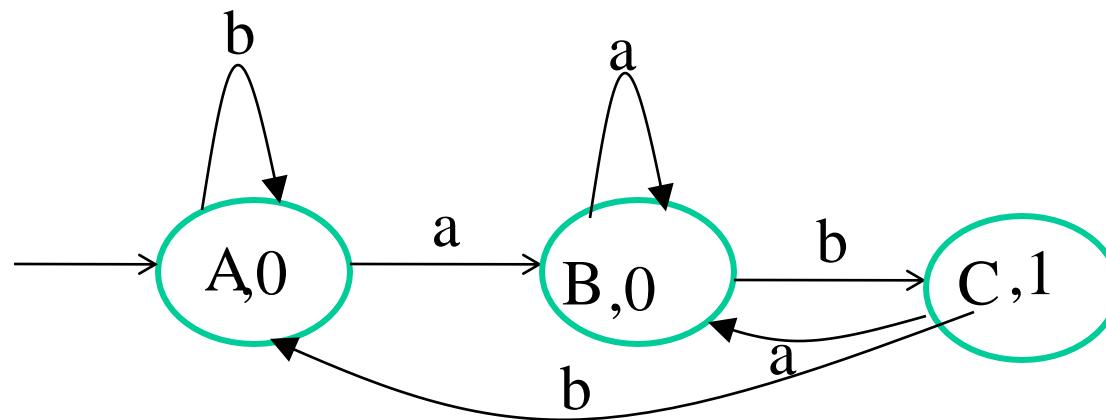


- Applying string 'ab' on machine (as input), got the output as 10.
- If number of input symbol is N then Number of output symbol is N.

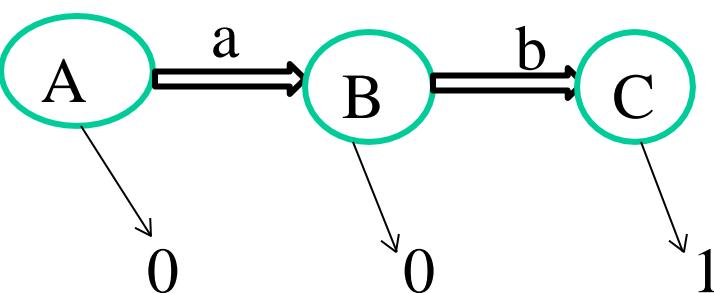
## Example on Moore Machine

Construct a Moore Machine that takes the set of all strings over  $\{a, b\}$  as i/p and prints '1' as o/p for every occurrence of 'ab' as a substring

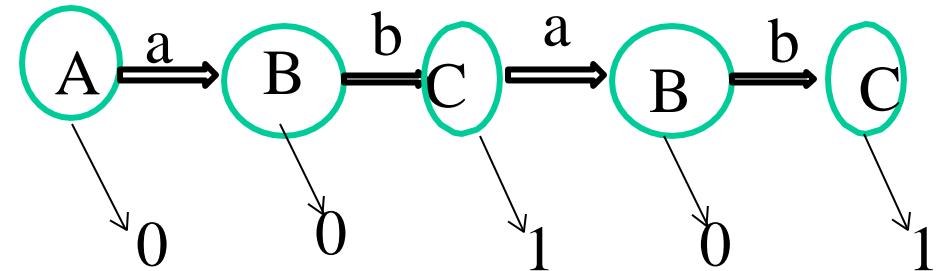
$$Q = \{A, B, C\} \quad \Sigma = \{a, b\} \quad \Delta = \{0, 1\}$$



• string 'ab' on machine



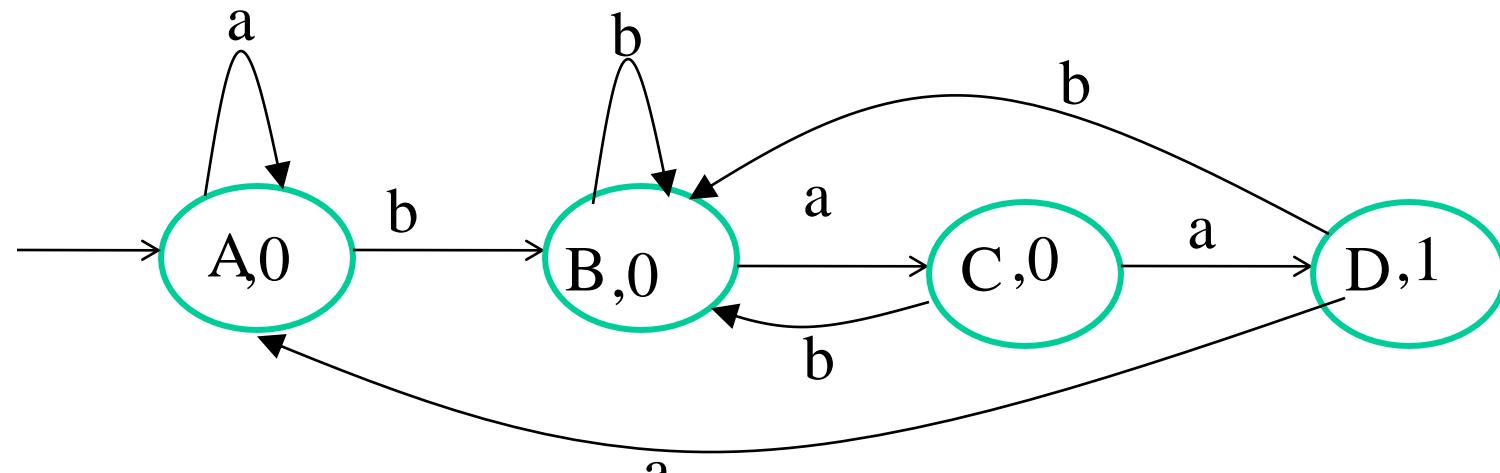
• string 'abab' on machine



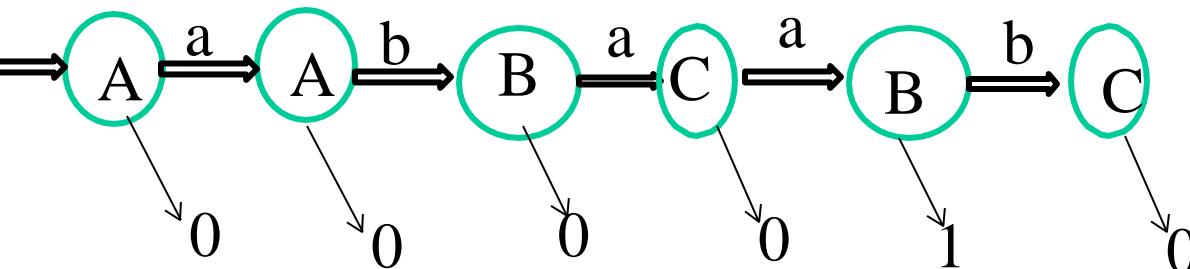
## Example on Moore Machine

Construct a Moore Machine that takes the set of all strings over  $\{a, b\}$  as i/p and counts no of occurrences of substring 'baa'

$$Q = \{A, B, C\} \quad \Sigma = \{a, b\} \quad \Delta = \{0, 1\}$$



• string 'abaab' on machine



## Example on Moore Machine

Construct a Moore Machine that takes the set of all strings over  $\{a, b\}$  as i/p and counts no of occurrences of substring 'baa'

$$Q = \{A, B, C, D\} \quad \Sigma = \{a, b\} \quad \Delta = \{0, 1\}$$

03/07/2019 Absent

- 5, 13, 21, 25, 32, 38, 42, 45, 48, 54, 62, 68, 71, 74

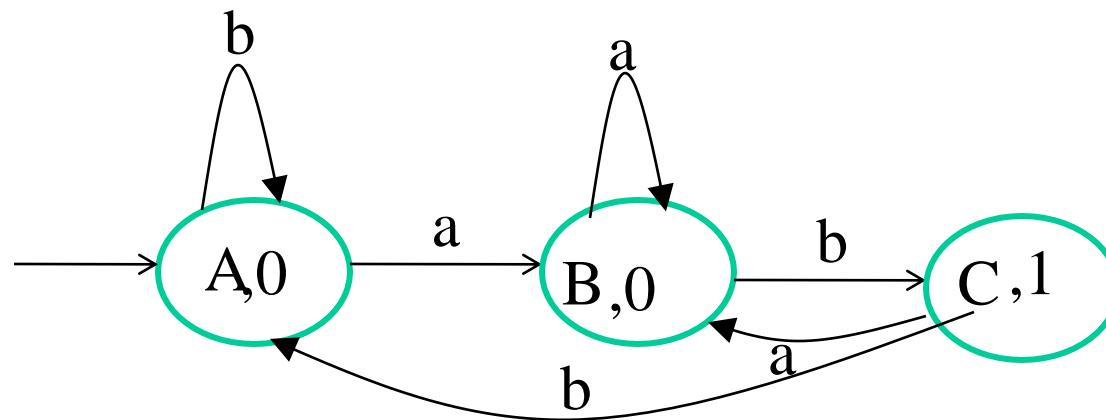
# Recall

- FA With O/P
- Moore Machine

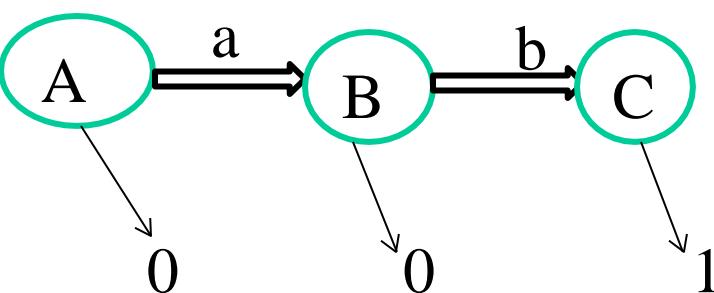
## Example on Moore Machine

Construct a Moore Machine that takes the set of all strings over  $\{a, b\}$  as i/p and prints '1' as o/p for every occurrence of 'ab' as a substring

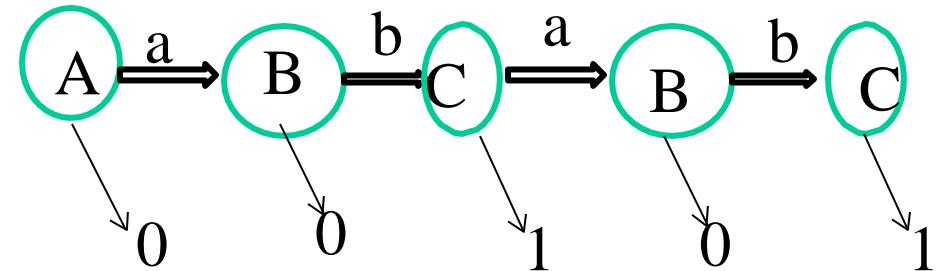
$$Q = \{A, B, C\} \quad \Sigma = \{a, b\} \quad \Delta = \{0, 1\}$$



• string 'ab' on machine



• string 'abab' on machine



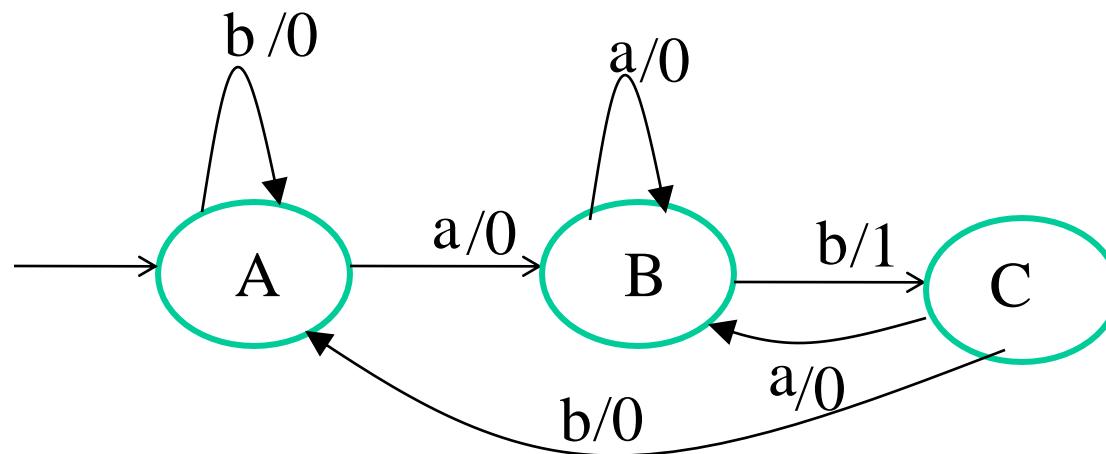
# Recall

- FA With O/P
- Moore Machine
- Mealy Machine

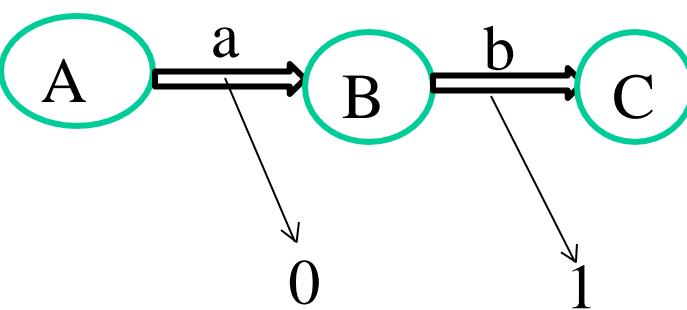
## Example on Mealy Machine

Construct a Mealy Machine that takes the set of all strings over  $\{a, b\}$  as i/p and prints '1' as o/p for every occurrence of 'ab' as a substring

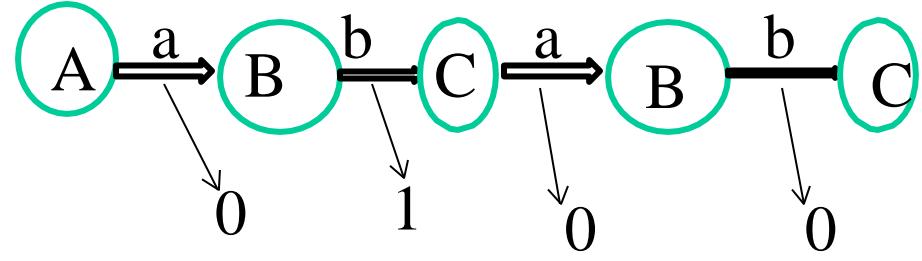
$$Q = \{A, B, C\}, \Sigma = \{a, b\}, \Delta = \{0, 1\}$$



• string 'ab' on machine

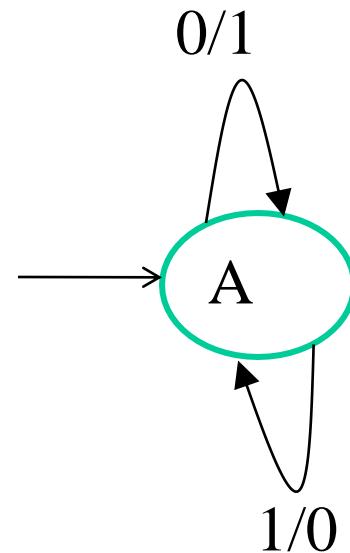


• string 'abab' on machine



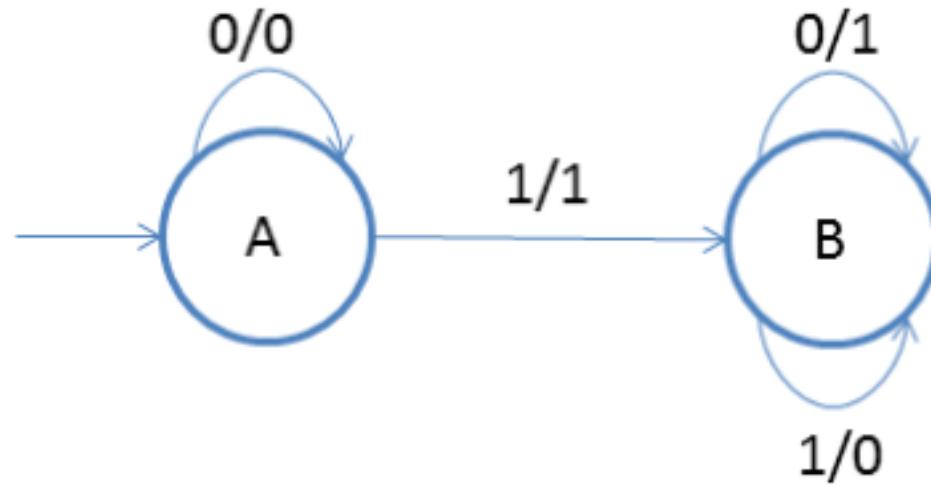
## Example on Mealy Machine

Construct a Mealy Machine for 1's complement of binary number.



## Example on Mealy Machine

Construct a Mealy Machine for 1's complement of binary number.





## Example on Mealy Machine

Construct a Mealy Machine that takes binary number as i/p and produces 2's complement of that number as o/p. Assume the string is read LSB to MSB and end carry is discarded.

## Example on Moore Machine

Construct a Moore Machine that takes the set of all strings over  $\{0, 1\}$  and produces 'A' as o/p if i/p ends with '10' or produces 'B' as o/p if i/p ends with '11' otherwise produces 'C'

## Example on Moore Machine

Construct a Moore Machine that takes binary no's as input and produces residue modulo '3' as output.

# THEORY OF COMPUTATION

---

Unit I

## FINITE STATE MACHINES

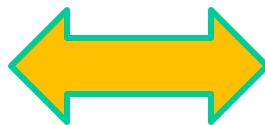
---

**Jagadish Kashinath Kamble** (M.Tech, IIT Kharagpur)

Dept. of Information Technology,  
Pune Institute of Computer Technology, Pune

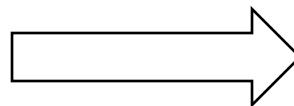
# Conversion of Moore and Mealy Machine

Moore Machine



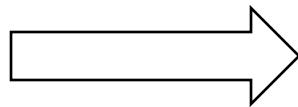
Mealy Machine

Moore Machine



Mealy Machine

Mealy Machine



Moore Machine

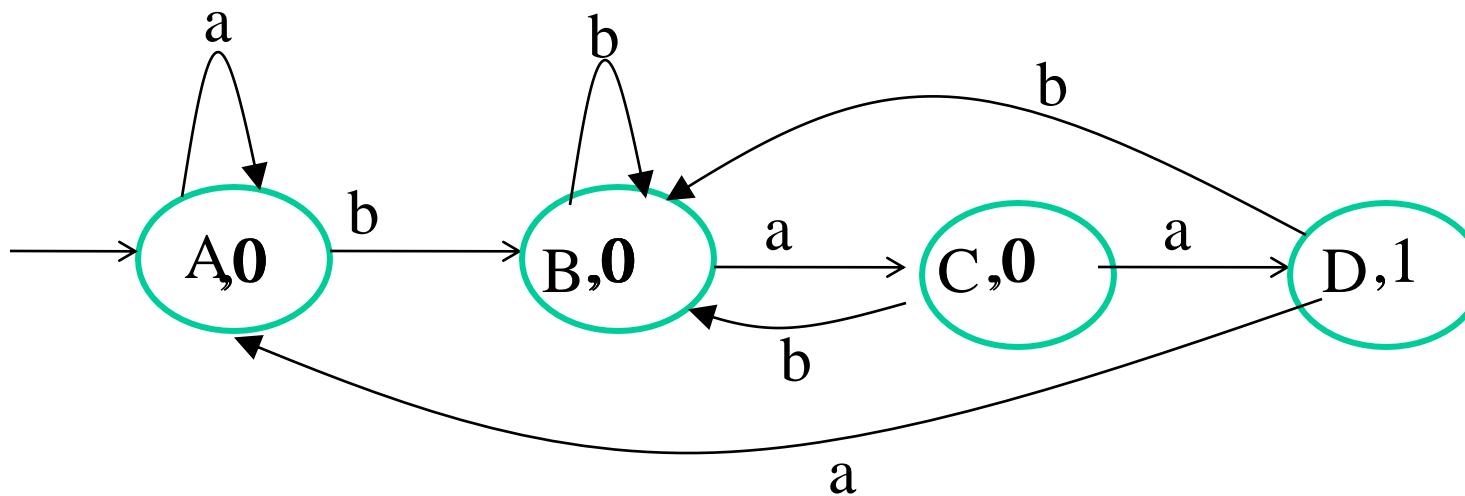
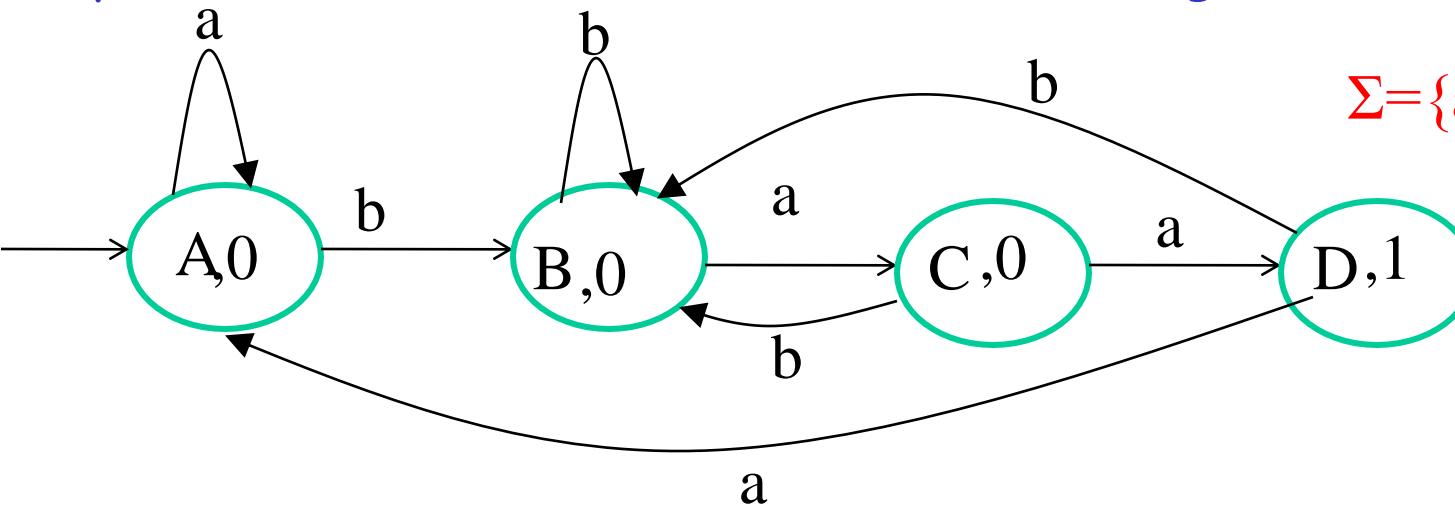
# Moore Machine to Mealy Machine

Construct a Moore Machine that takes the set of all strings over  $\{a, b\}$  as i/p and counts no of occurrences of substring 'baa'

$$Q = \{A, B, C\}$$

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

$$\Delta = \{0, 1\}$$



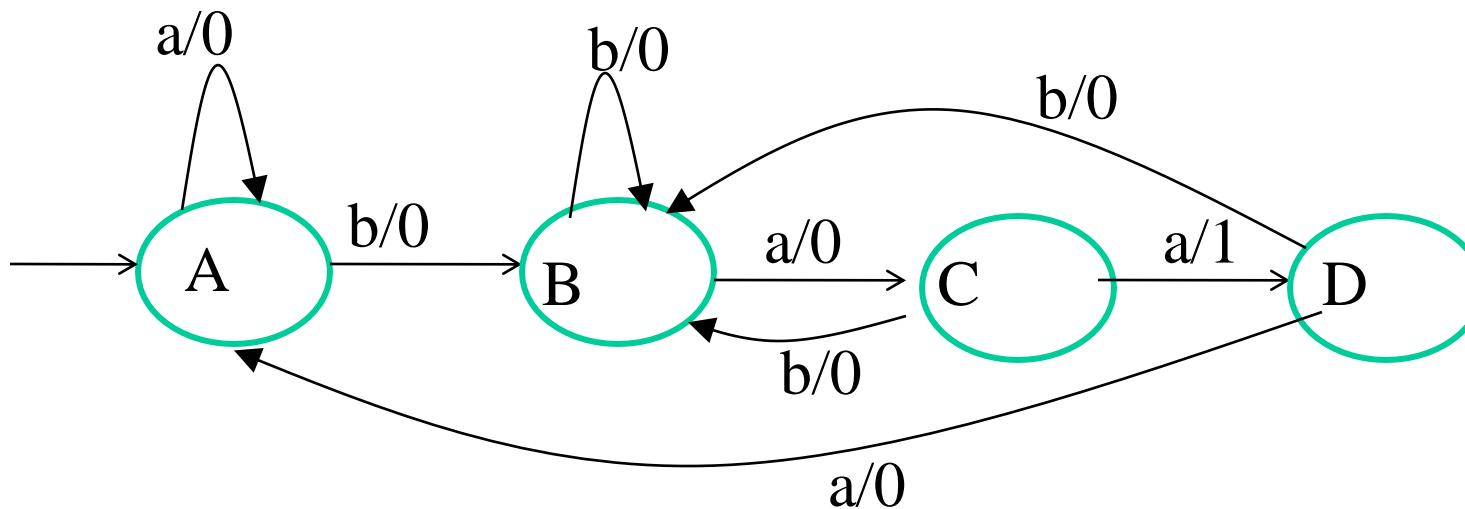
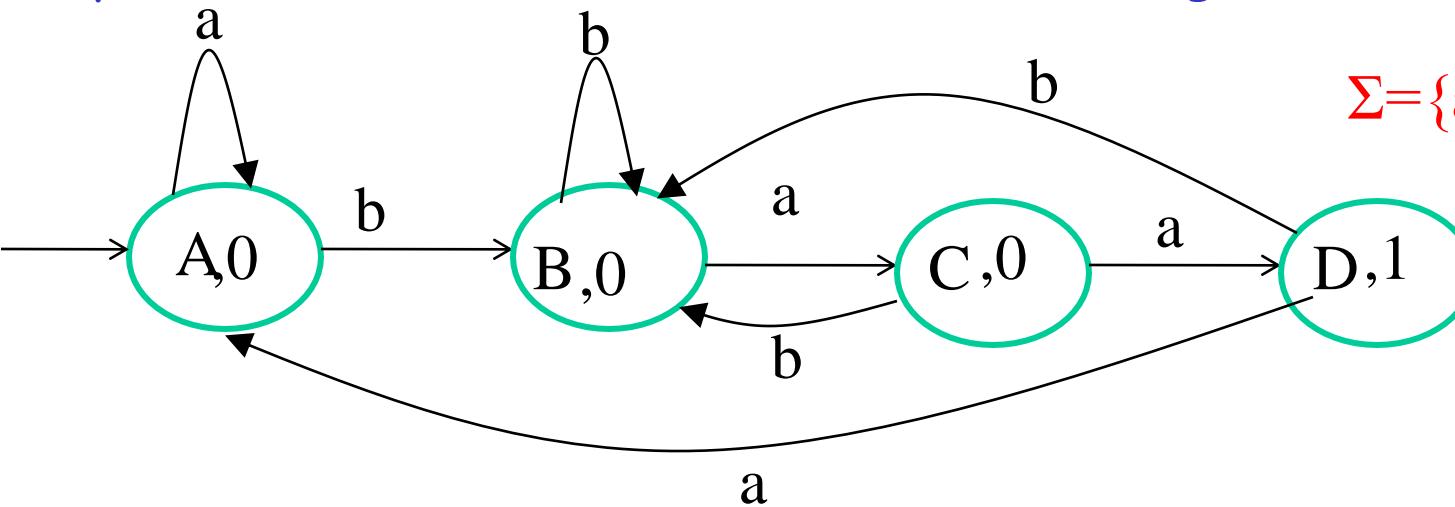
# Moore Machine to Mealy Machine

Construct a Moore Machine that takes the set of all strings over  $\{a, b\}$  as i/p and counts no of occurrences of substring 'baa'

$$Q = \{A, B, C\}$$

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

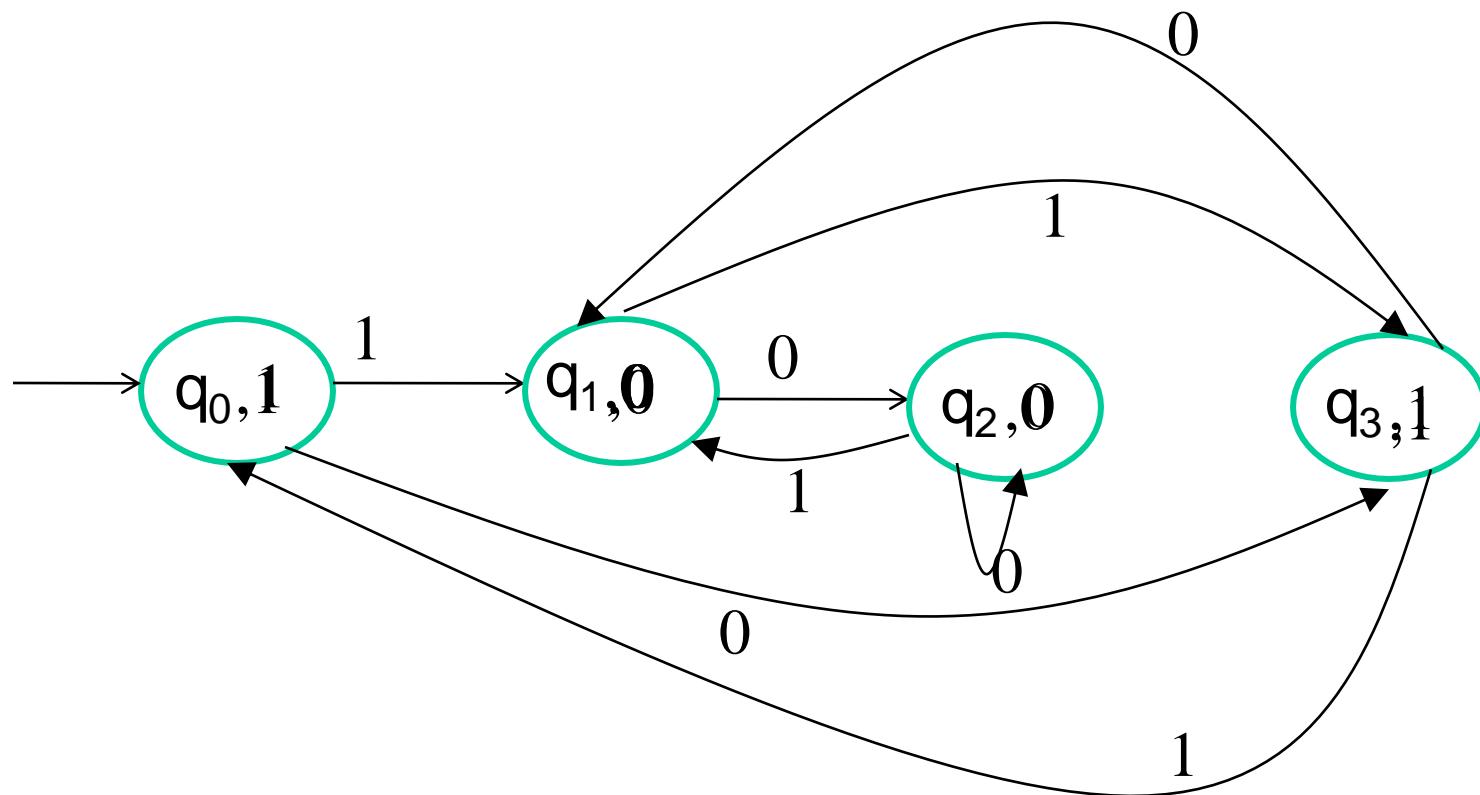
$$\Delta = \{0, 1\}$$



# Convert Following Moore Machine into Mealy Machine

Present State	Next State		Output
	$a = 0$	$a = 1$	
$\rightarrow q_0$	$q_3$	$q_1$	1
$q_1$	$q_2$	$q_3$	0
$q_2$	$q_2$	$q_1$	0
$q_3$	$q_1$	$q_0$	1

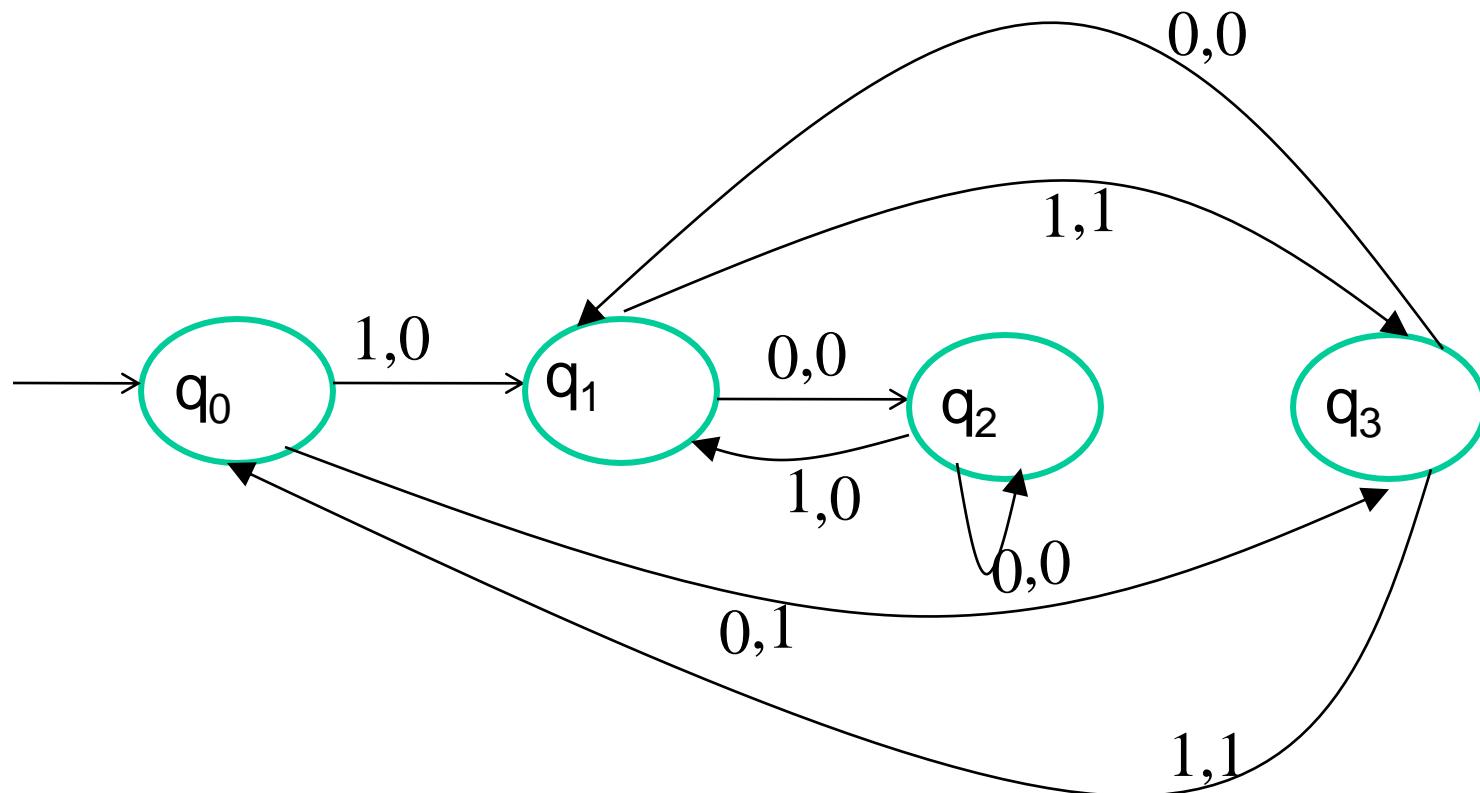
Aug-2015  
INSEM



# Convert Following Moore Machine into Mealy Machine

Present State	Next State		Output
	$a = 0$	$a = 1$	
$\rightarrow q_0$	$q_3$	$q_1$	1
$q_1$	$q_2$	$q_3$	0
$q_2$	$q_2$	$q_1$	0
$q_3$	$q_1$	$q_0$	1

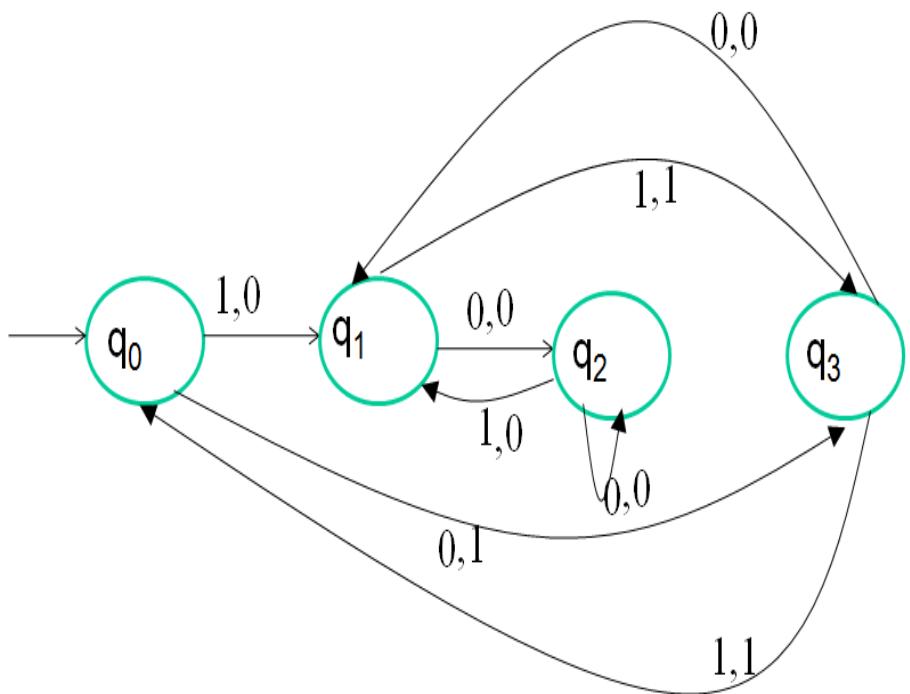
Aug-2015  
INSEM



# Convert Following Moore Machine into Mealy Machine

Present State	Next State		Output
	$a = 0$	$a = 1$	
$\rightarrow q_0$	$q_3$	$q_1$	1
$q_1$	$q_2$	$q_3$	0
$q_2$	$q_2$	$q_1$	0
$q_3$	$q_1$	$q_0$	1

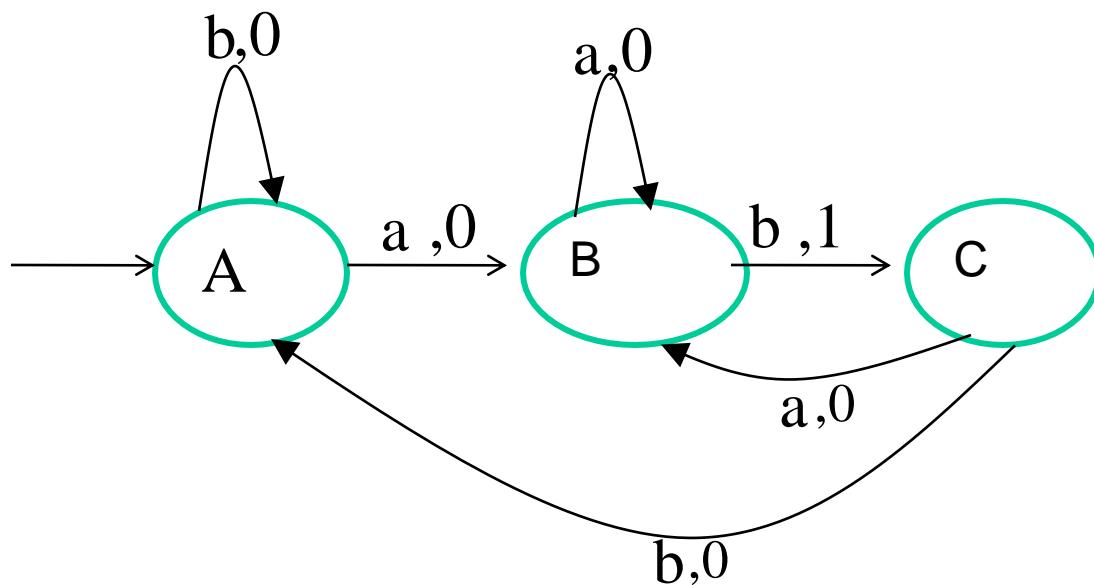
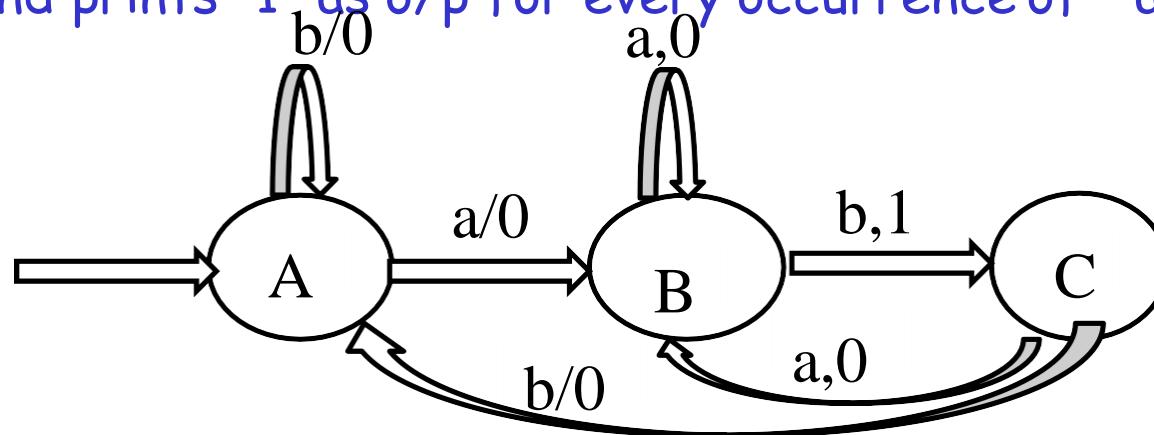
Aug-2015  
INSEM



	Input	Output	Input	Output
	0		1	
$q_0$	$q_3$	1	$q_1$	0
$q_1$	$q_2$	0	$q_3$	1
$q_2$	$q_2$	0	$q_1$	0
$q_3$	$q_1$	0	$q_0$	1

# Mealy Machine to Moory Machine

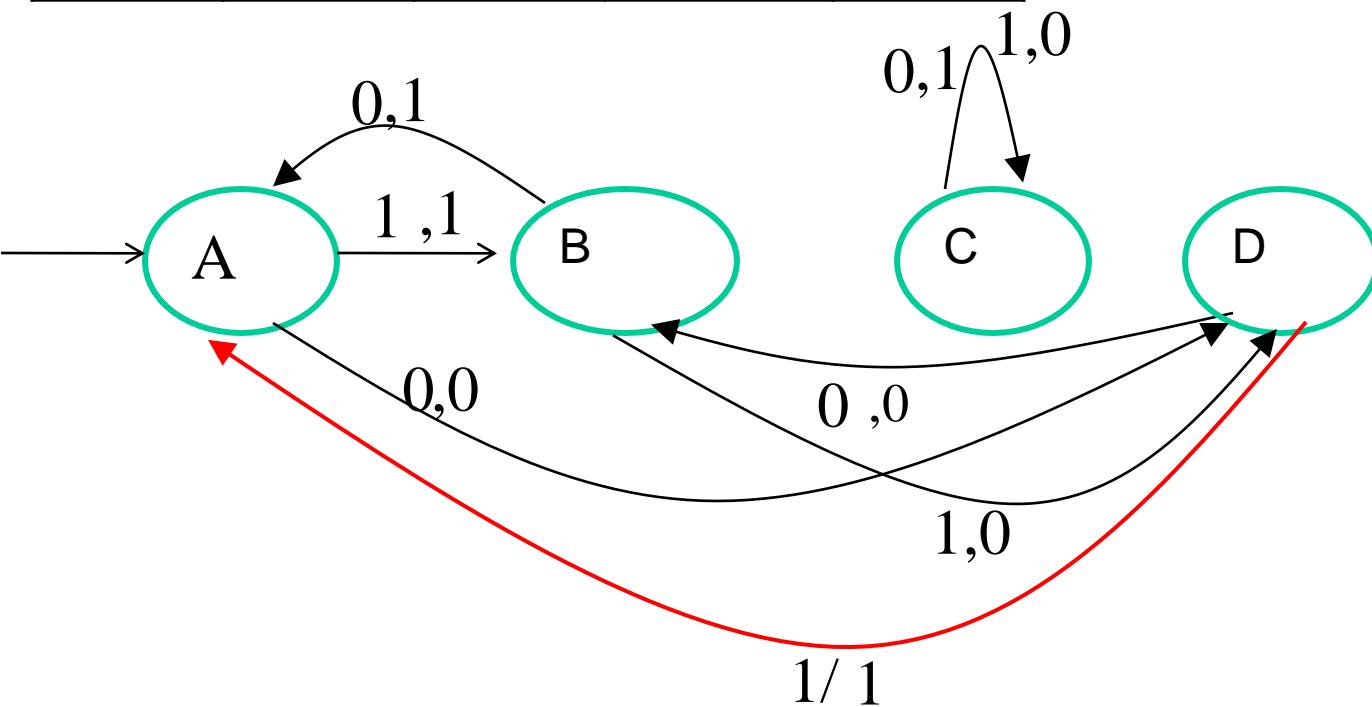
Construct a Mealy Machine that takes the set of all strings over  $\{a, b\}$  as i/p and prints '1' as o/p for every occurrence of 'ab' as a substring

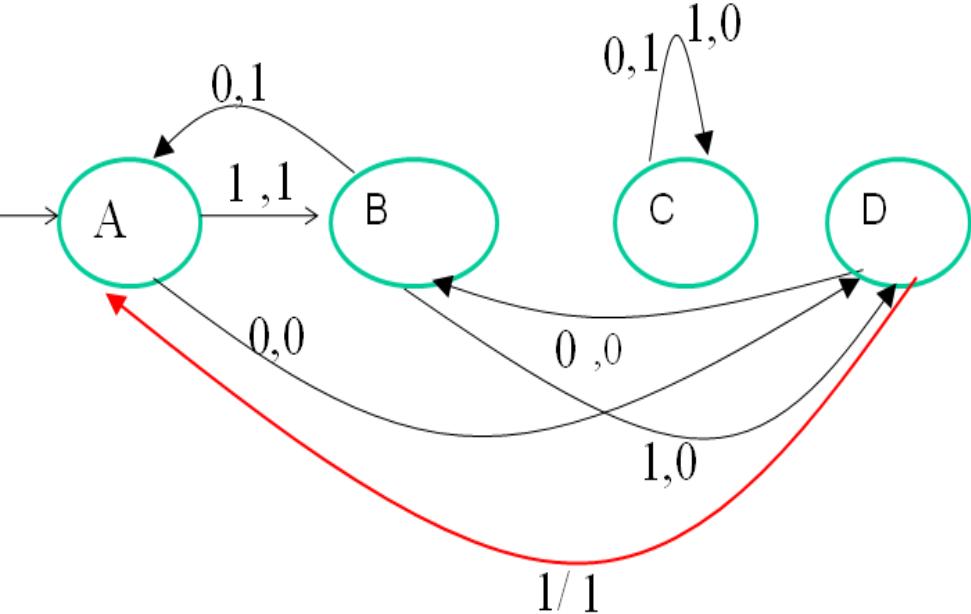


# Convert Following Mealy Machine into Moore Machine

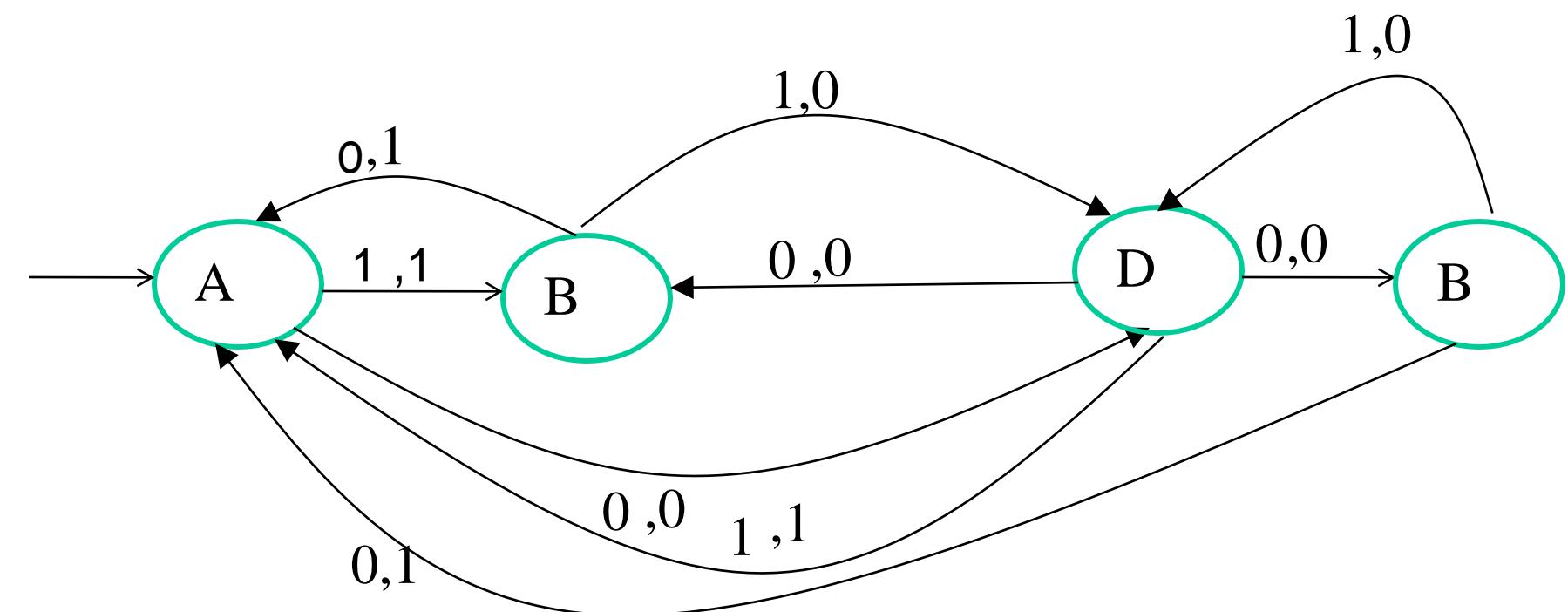
Present State	0		1	
	Next State	Output	Next State	Output
→ A	D	0	B	1
B	A	1	D	0
C	C	1	C	0
D	B	0	A	1

Aug-2015  
INSEM





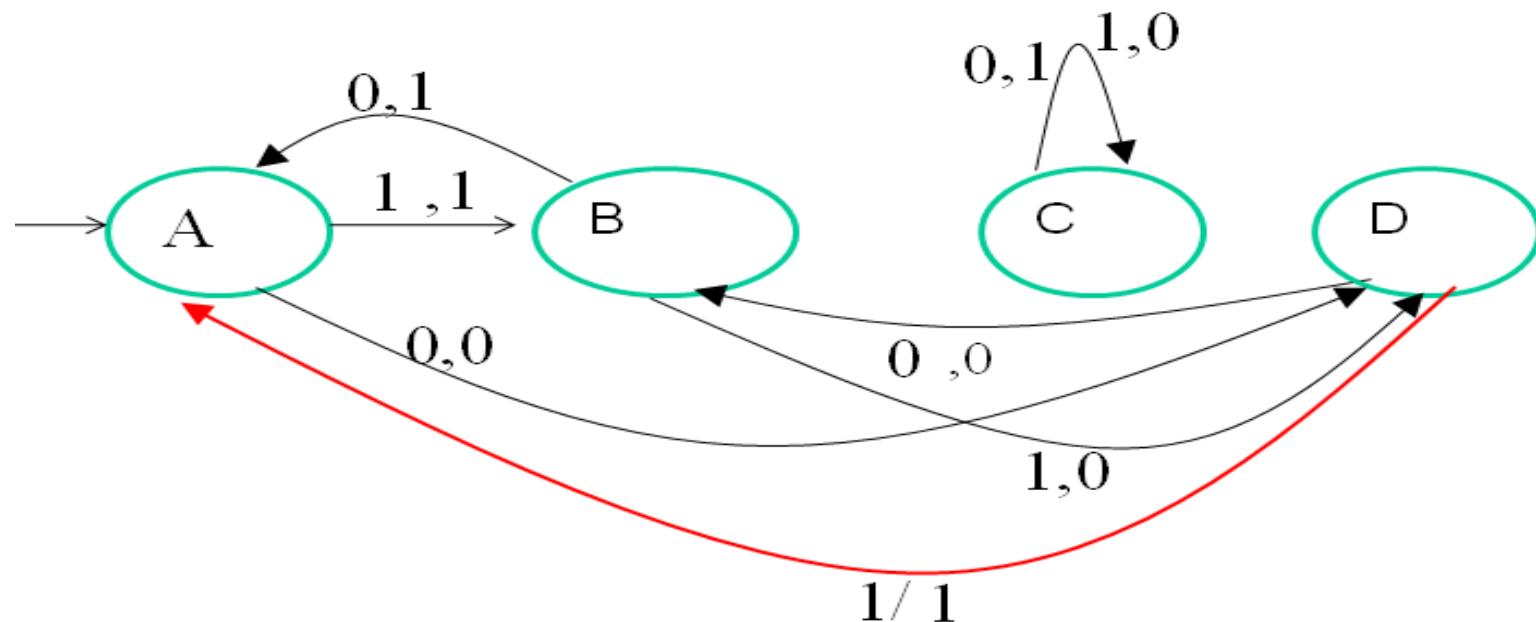
Present State	0		1	
	Next State	Output	Next State	Output
$\rightarrow A$	D	0	B	1
B	A	1	D	0
C	C	1	C	0
D	B	0	A	1



# Convert Following Mealy Machine into Moore Machine

Present State	0		1	
	Next State	Output	Next State	Output
→ A	D	0	B	1
B	A	1	D	0
C	C	1	C	0
D	B	0	A	1

Aug-2015  
INSEM



Present State	0		1	
	Next State	Output	Next State	Output
→ A	D	0	B	1
B	A	1	D	0
C	C	1	C	0
D	B	0	A	1

Present State	Input		Output
	0	1	
A	D	B1	1
D	B0	A	0
B0	A	D	0
B1	A	D	1
C0	C1	C0	0
C1	C1	C0	1

b) Construct Mealy machine equivalent to the given Moore machine

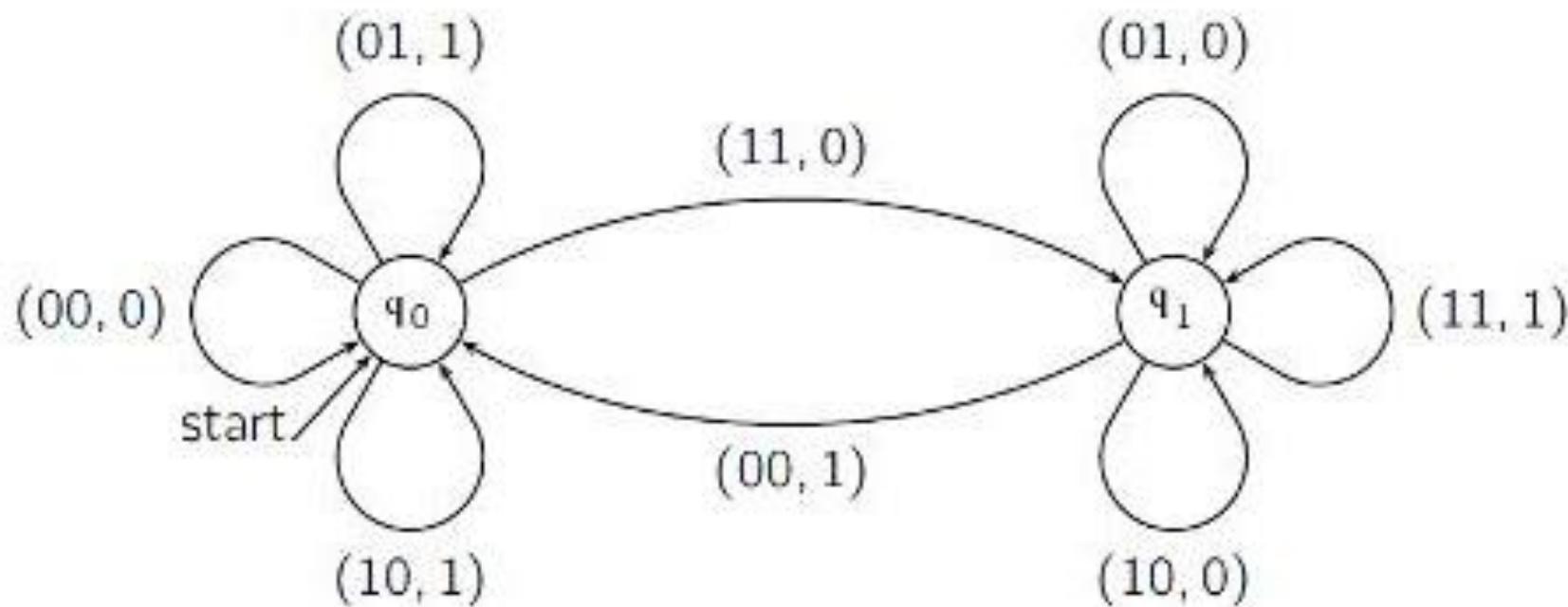
Aug-2017

INSEM

	0	1	O/P
q0	q0	q1	N
q1	q0	q2	N
q2	q0	q3	N
q3	q0	q3	Y

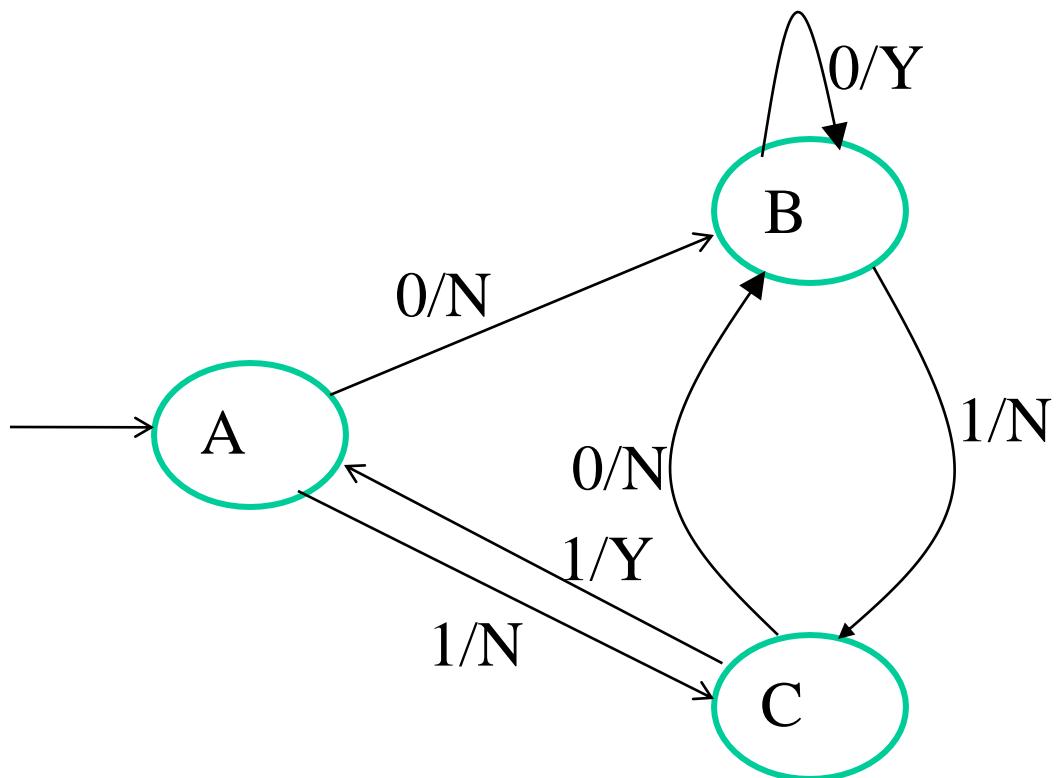
Start state : q0 ; Final state : q3

Design a finite automata which perform addition of two Binary number.



Design Moore Machine for divisibility by 3  
tester for binary number. (Nov-2017 6 Marks)

Convert following Mealy Machine to Moore  
Machine (Nov-2017 6 Marks)



04/07/2019Absent

- 1,13,22,27,28,38,42,45,48,51/,61,71

- University Question Solving Session

05/07/2019Absent

- 32,38,45,50,71,74

Thank you!!!!!!

# Formal Definition

Deterministic Finite Automaton (DFA)

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

$Q$  : set of states

$\Sigma$  : input alphabet       $\lambda \notin \Sigma$

$\delta$  : transition function

$q_0$  : initial state

$F$  : set of accepting states

# Alphabets and Strings

An alphabet is a set of symbols

Example Alphabet:  $\Sigma = \{a, b\}$

A string is a sequence of symbols from the alphabet

Example Strings

$a$	$u = ab$
$ab$	$v = bbbaaa$
$abba$	$w = abba$
$aaabbbaaba$	$b$

Decimal numbers alphabet  $\Sigma = \{0,1,2,\dots,9\}$

102345

567463386

Binary numbers alphabet  $\Sigma = \{0,1\}$

100010001

101101111

Unary numbers alphabet       $\Sigma = \{1\}$

Unary number: 1      11      111      1111      11111

Decimal number: 1      2      3      4      5

# String Operations

$w = a_1 a_2 \cdots a_n$

$abba$

$v = b_1 b_2 \cdots b_m$

$bbbaaa$

## Concatenation

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

$abbabbbaaa$

$w = a_1 a_2 \cdots a_n$  $ababa aabb b$ 

Reverse

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

String Length  
 $w = a_1 a_2 \cdots a_n$

$$|w| = n$$

Length:

$$|abba| = 4$$

Examples:

$$|aa| = 2$$

$$|a| = 1$$

# Empty String

$\lambda$  or  $\epsilon$

A string with no letters is denoted:

$$|\lambda| = 0$$

Observations:

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

$$\lambda abba = abba \lambda = ab\lambda ba = abba$$

# Substring

Substring of string:

a subsequence of consecutive characters

String

abbab

Substring

*ab*

abbab

*abba*

abbab

*b*

abbab

*bbab*

# Prefix and Suffix

*abbab*

Prefixes      Suffixes

$\lambda$

*abbab*

*a*

*bbab*

*ab*

*bab*

*abb*

*ab*

*abba*

*b*

*abbab*

$\lambda$

$w = uv$

prefix

suffix

The diagram shows the string  $w = uv$  at the top. Below it, the word "prefix" is written in blue, with an arrow pointing from it to the first part of the string  $uv$ . Below "prefix", the word "suffix" is written in blue, with an arrow pointing from it to the second part of the string  $uv$ .

Another Operation

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

$$(abba)^2 = abbaabba$$

Example:

$$w^0 = \lambda$$

Definition:

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

# The \* Operation

$\Sigma^*$ : the set of all possible strings from alphabet

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

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

# The + Operation

$\Sigma^+$  : the set of all possible strings from alphabet  $\Sigma$  except  $\lambda$

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

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

$$\Sigma^+ = \Sigma^* - \lambda$$

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

# Languages

**Language:** a set of strings

In mathematics, computer science, and linguistics, a formal language consists of words whose letters are taken from an alphabet and are well-formed according to a specific set of rules.

**String:** a sequence of symbols

from some alphabet

Example: Alphabet:  $\Sigma = \{a, b, c, \dots, z\}$

Strings: cat, dog, house

Language: {cat, dog, house}

A language over alphabet  $\Sigma$   
is any subset of  $\Sigma^*$

Examples:

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

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

Language:  $\{\lambda\}$

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

Language:  $\{\lambda, abba, baba, aa, ab, aaaaaaa\}$

Languages are used to describe computation problems:

$$PRIMES = \{2, 3, 5, 7, 11, 13, 17, \dots\}$$

$$EVEN = \{0, 2, 4, 6, \dots\}$$

Alphabet:  $\Sigma = \{0, 1, 2, \dots, 9\}$

# More Language Examples

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

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

$\lambda$

$ab$

$aabb$

$aaaaabbbbb$

$\} \in L$

$abb \notin L$

# Prime numbers

Alphabet  $\Sigma = \{0,1,2,\dots,9\}$

Language:

$PRIMES = \{x : x \in \Sigma^* \text{ and } x \text{ is prime}\}$

$PRIMES = \{2,3,5,7,11,13,17,\dots\}$

# Even and odd numbers

Alphabet  $\Sigma = \{0,1,2,\dots,9\}$

$EVEN = \{x : x \in \Sigma^* \text{ and } x \text{ is even}\}$

$EVEN = \{0,2,4,6,\dots\}$

$ODD = \{x : x \in \Sigma^* \text{ and } x \text{ is odd}\}$

$ODD = \{1,3,5,7,\dots\}$

# Unary Addition

Alphabet:  $\Sigma = \{1, +, =\}$

Language:

$ADDITION = \{x + y = z : x = 1^n, y = 1^m, z = 1^k,$   
 $n + m = k\}$

$11 + 111 = 11111 \in ADDITION$

$111 + 111 = 111 \notin ADDITION$

Note that:

Sets

$$\emptyset = \{ \ } \neq \{\lambda\}$$

Set size

$$|\{ \ }| = |\emptyset| = 0$$

Set size

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

String length

$$|\lambda| = 0$$

# Formal Definition

Deterministic Finite Automaton (DFA)

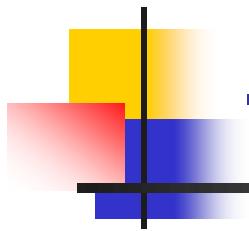
$Q$  : set of states

$\Sigma$  : input alphabet       $\lambda \notin \Sigma$

$\delta$  : transition function

$q_0$  : initial state

$F$  : set of accepting states



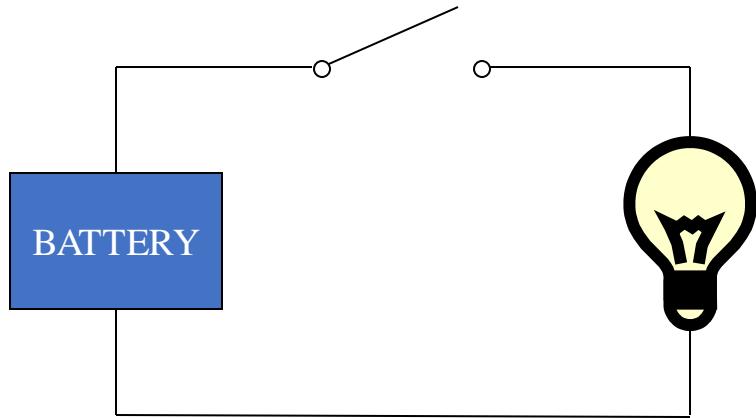
# Deterministic Finite Automata

## - Definition

- A Deterministic Finite Automaton (DFA) consists of:
  - $Q \implies$  a finite set of states
  - $\Sigma \implies$  a finite set of input symbols (alphabet)
  - $q_0 \implies$  a start state
  - $F \implies$  set of accepting states
  - $\delta \implies$  a transition function, which is a mapping between  $Q \times \Sigma \implies Q$
- A DFA is defined by the 5-tuple:
  - $\{Q, \Sigma, q_0, F, \delta\}$

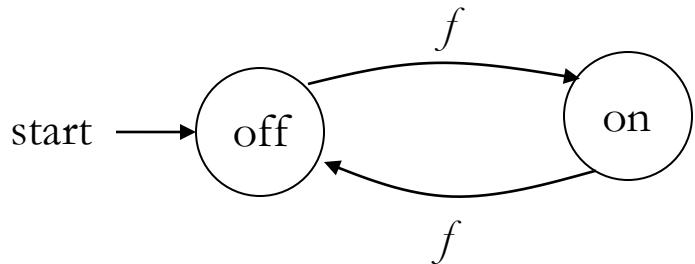
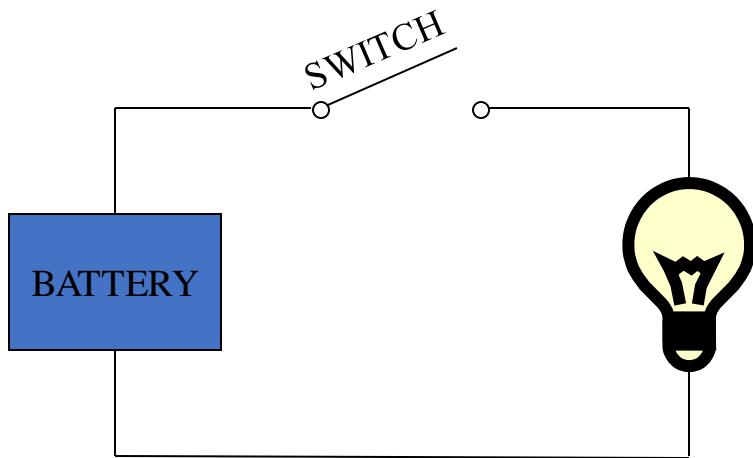
# A simple “computer”

---



# A simple “computer”

---



**input:** switch

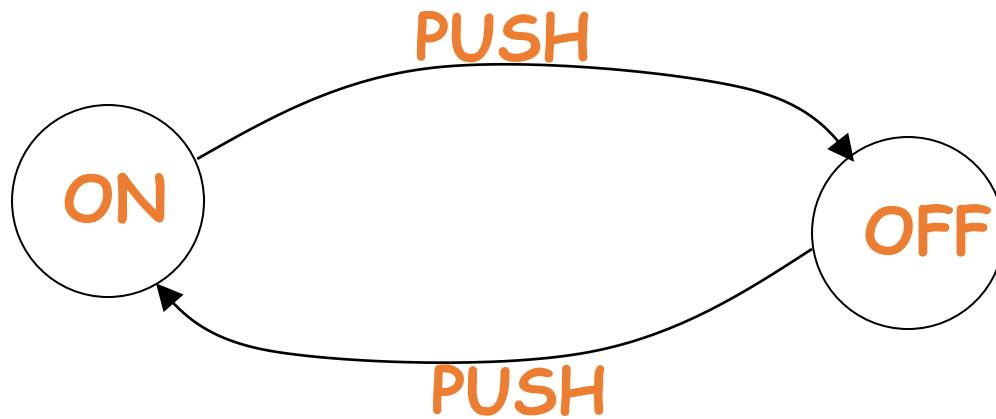
**output:** light bulb

**actions:**  $f$  for “flip switch”

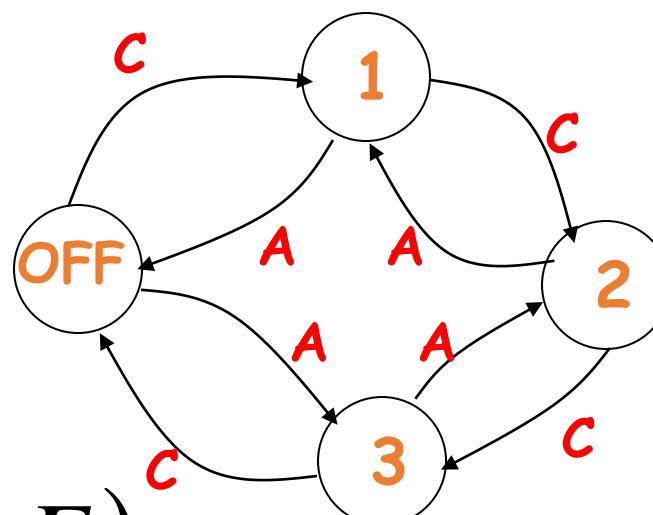
**states:** on, off

bulb is on if and only if  
there was an odd number  
of flips

• Electric  
Switch



• FAN  
REGULATOR



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

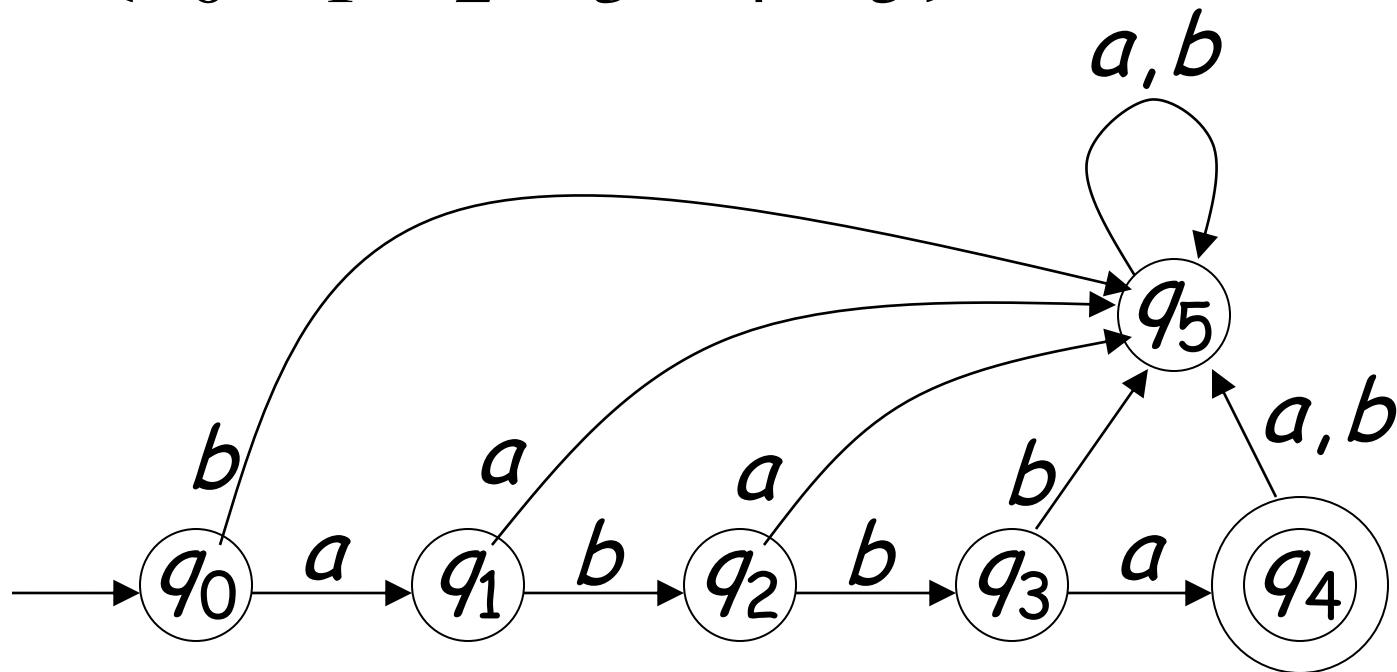
$$Q = \{on, off\} \quad Q = \{on, off\}$$

$Q$ 

# Set of States

## Example

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



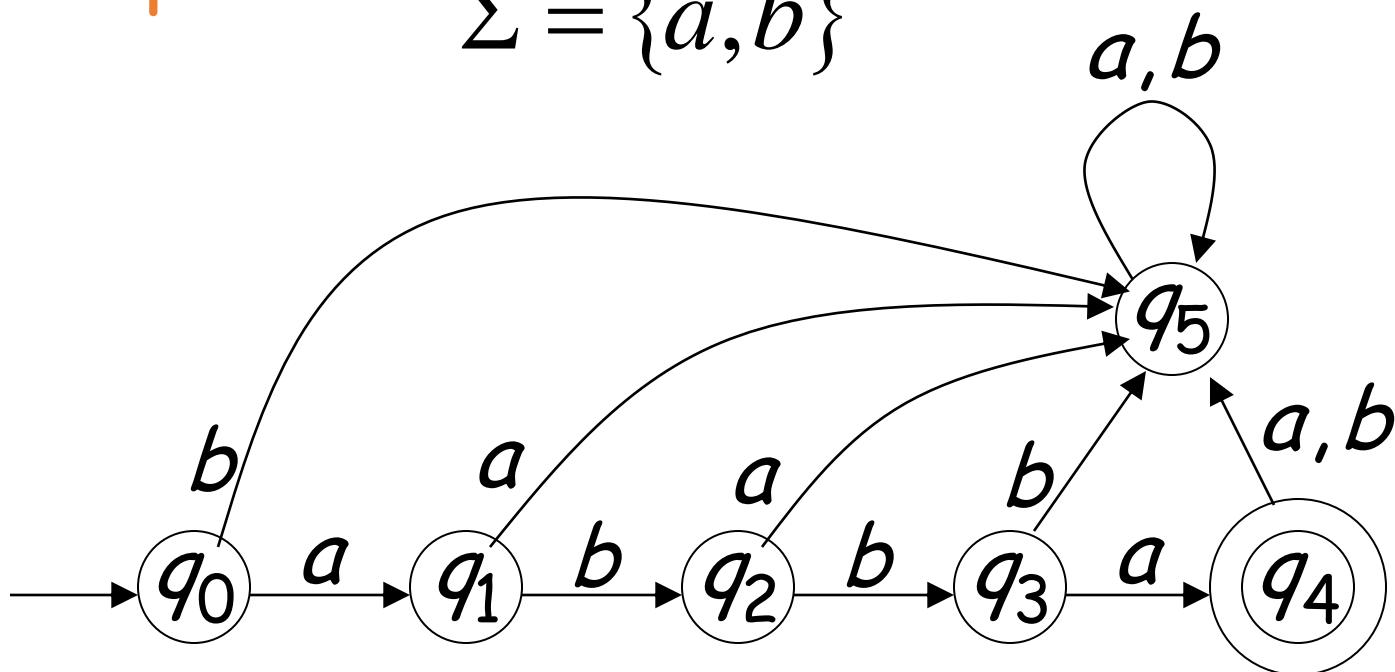
$\Sigma$ 

Input Alphabet

$\lambda \notin \Sigma$  : the input alphabet never contains  $\lambda$

Example

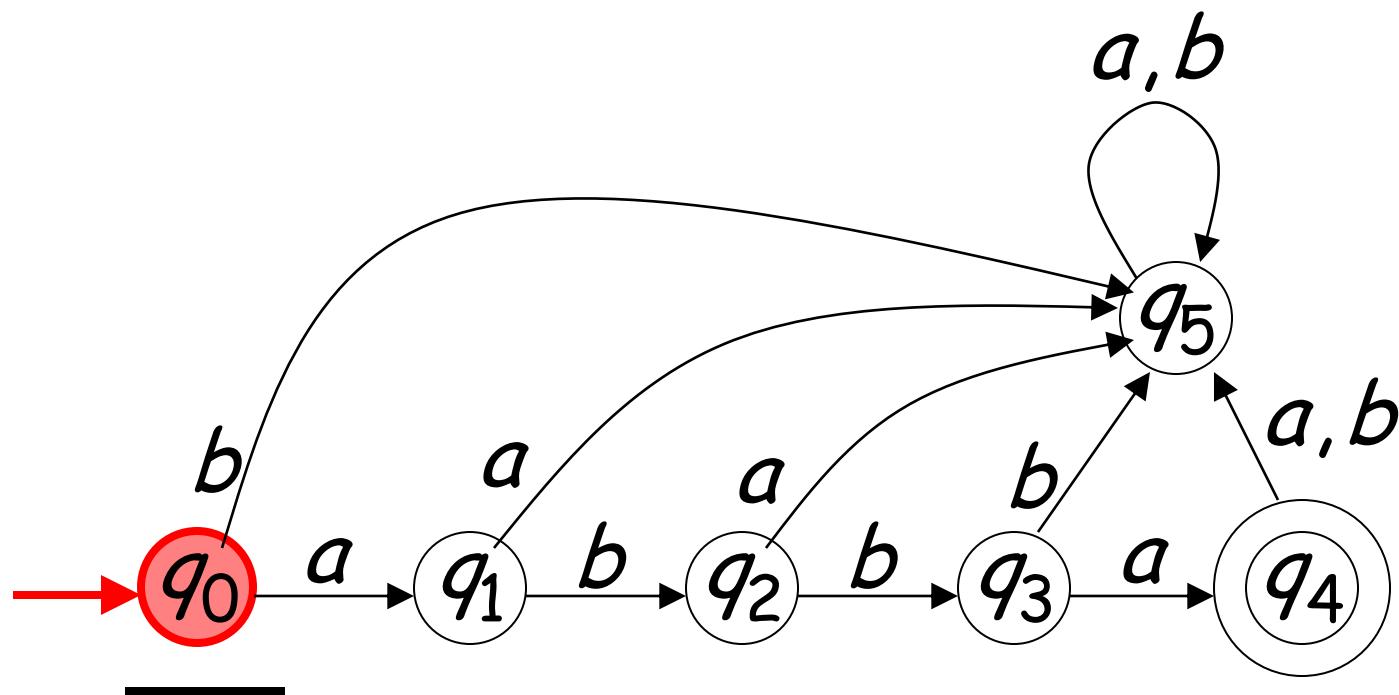
$$\Sigma = \{a, b\}$$



$q_0$ 

Initial State

Example

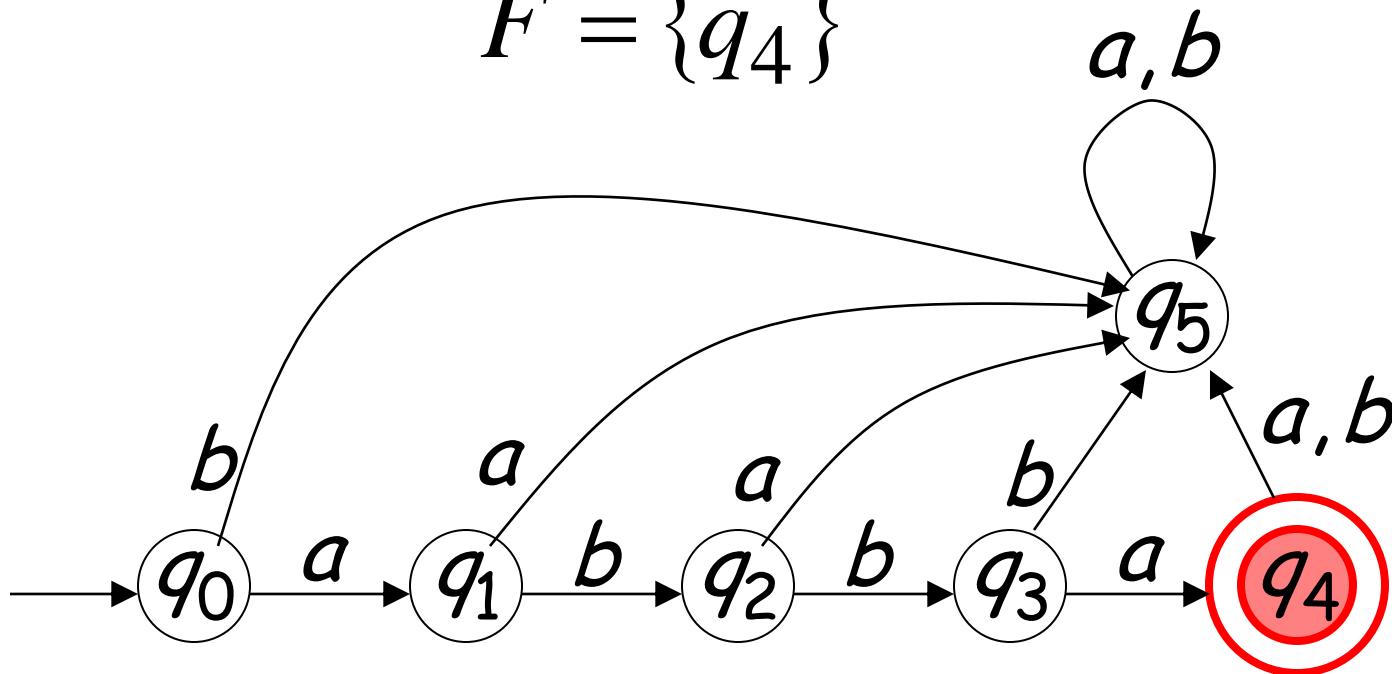


$$F \subseteq Q$$

Set of Accepting States

Example

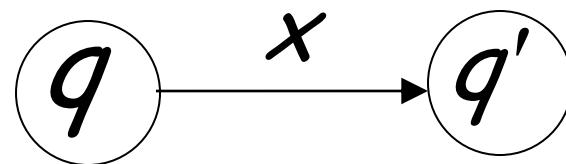
$$F = \{q_4\}$$



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

Transition Function

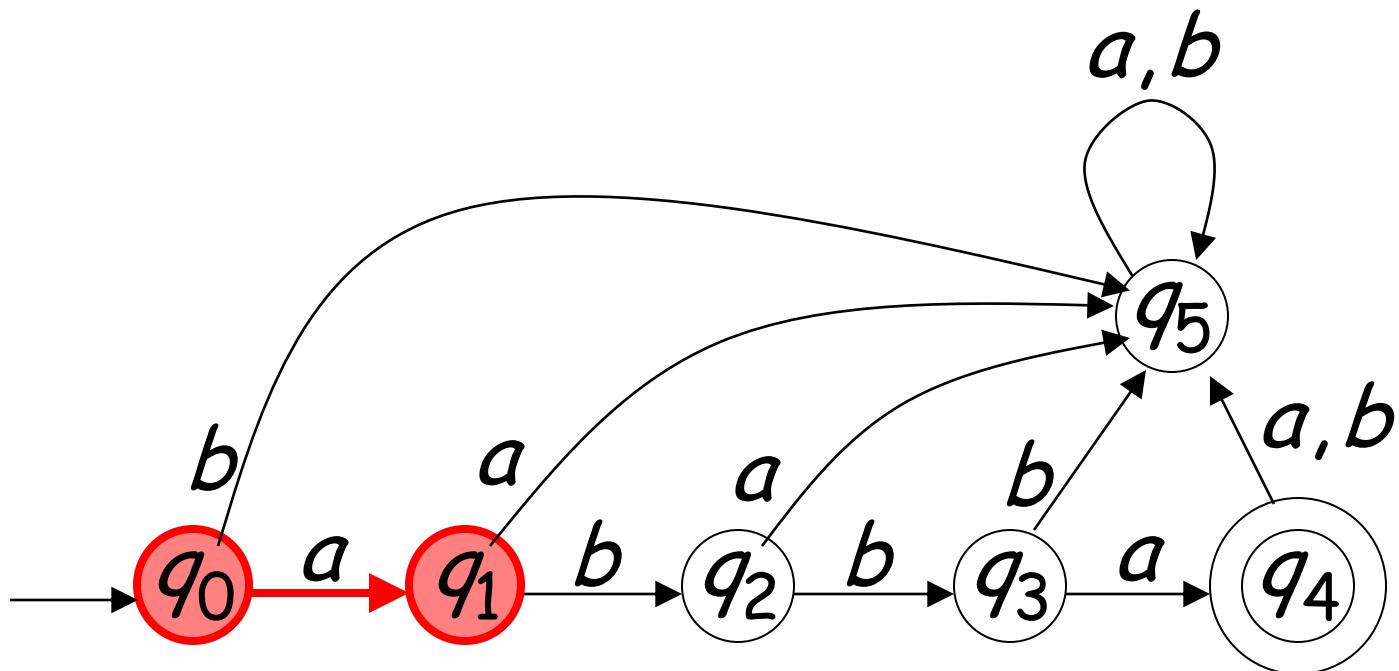
$$\delta(q, x) = q'$$



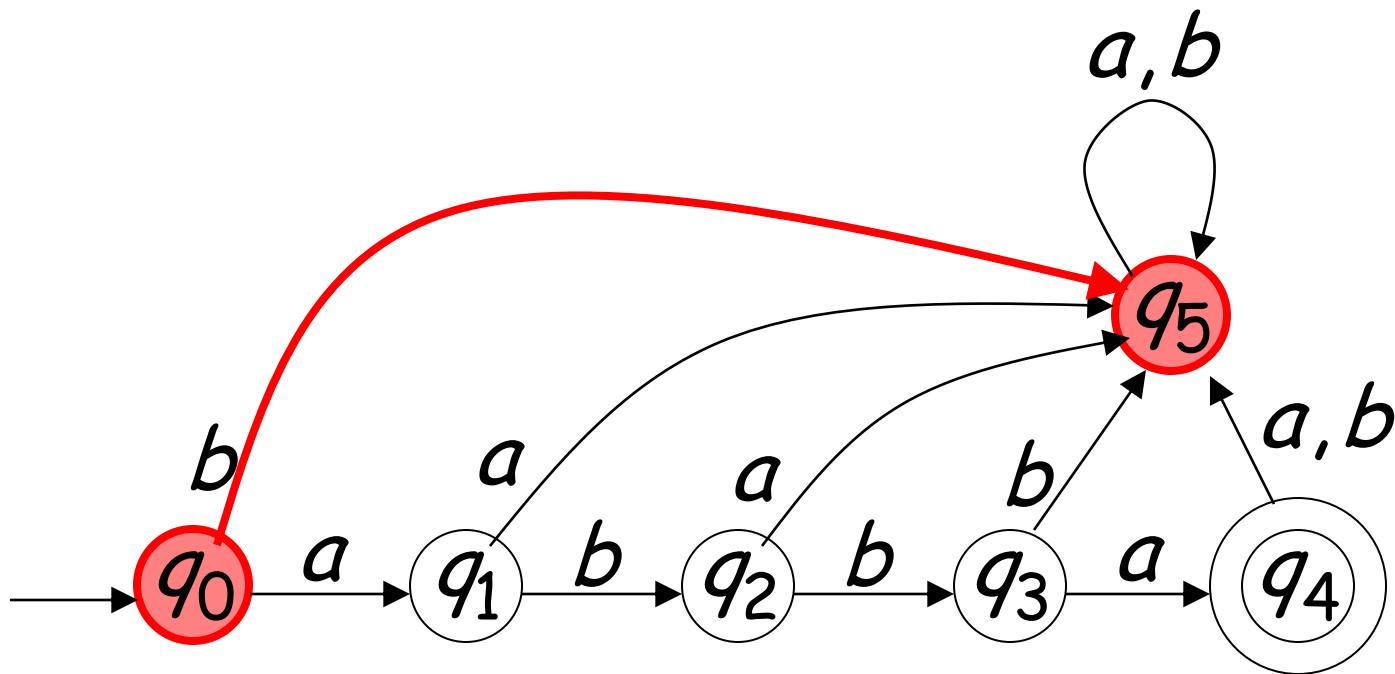
Describes the result of a transition  
from state  $q$  with symbol  $x$

Example:

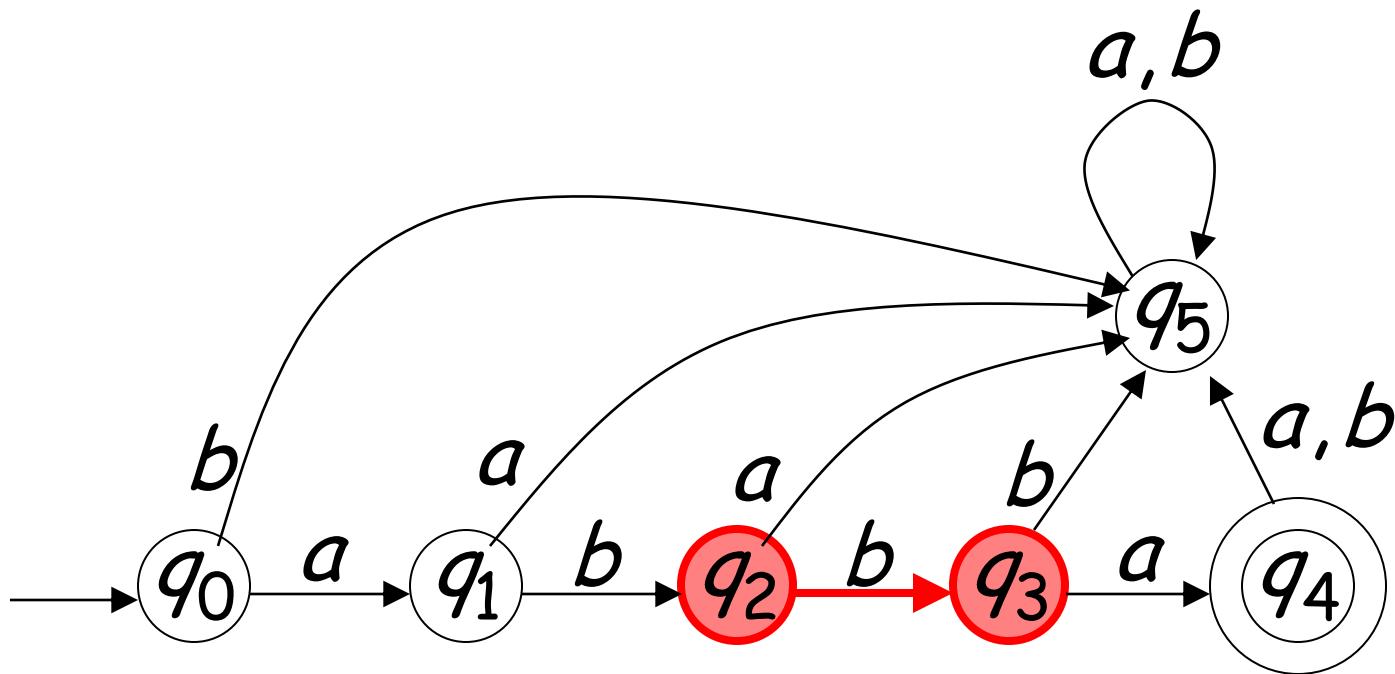
$$\delta(q_0, a) = q_1$$



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



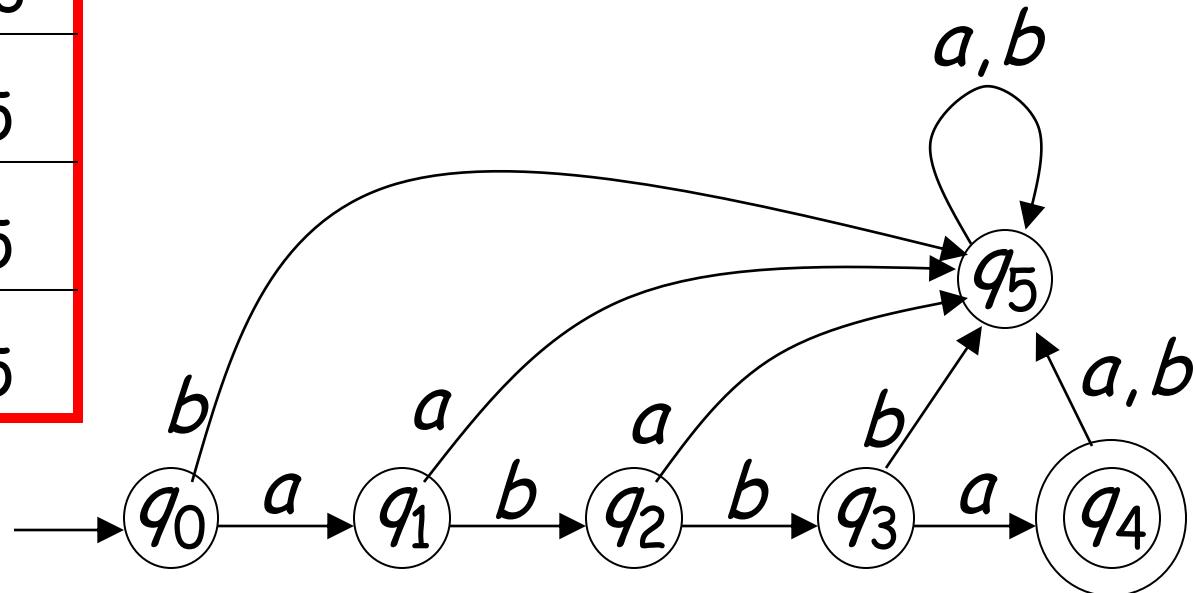
$$\delta(q_2, b) = q_3$$



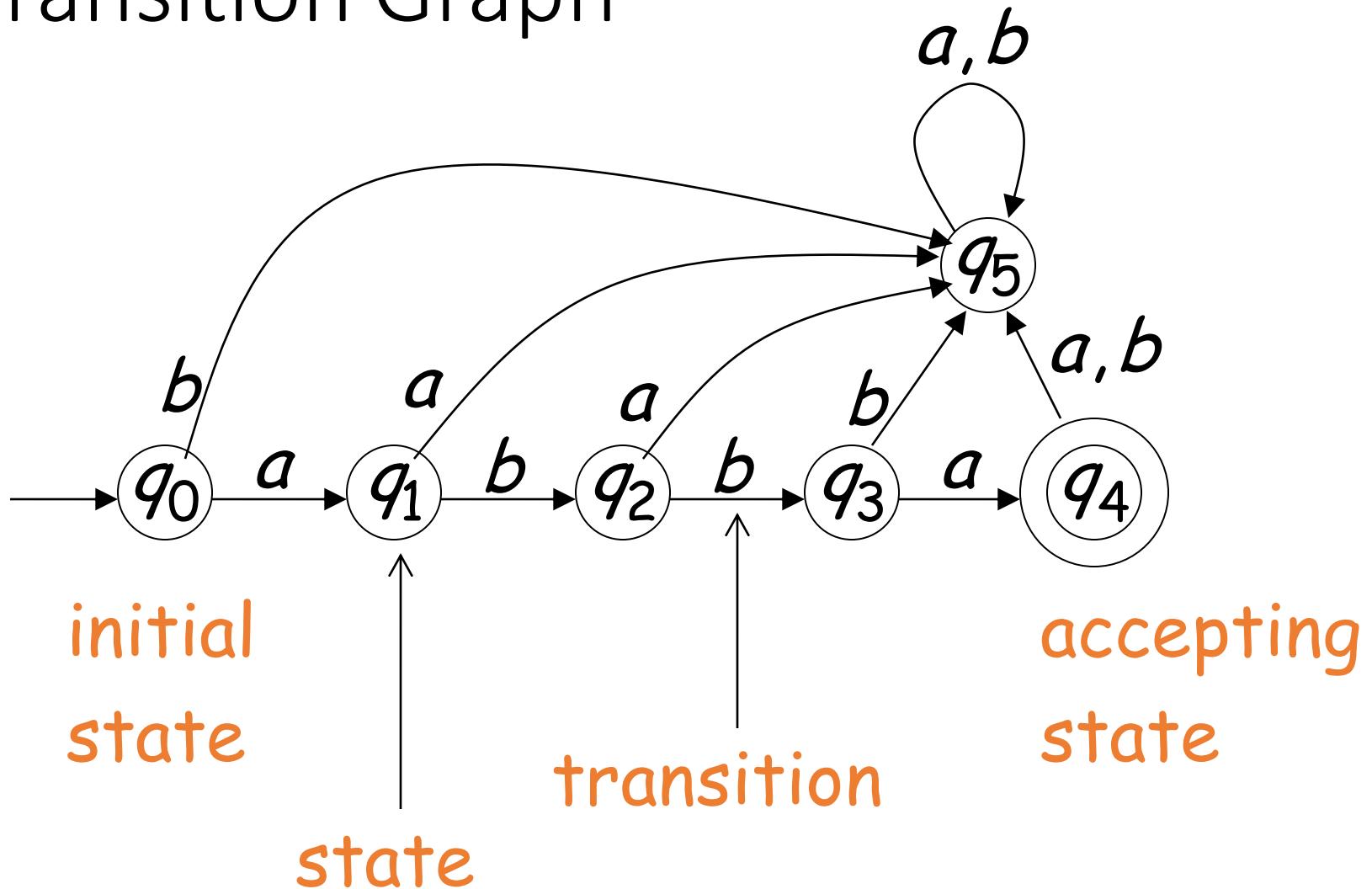
# Transition Table for $\delta$

symbols

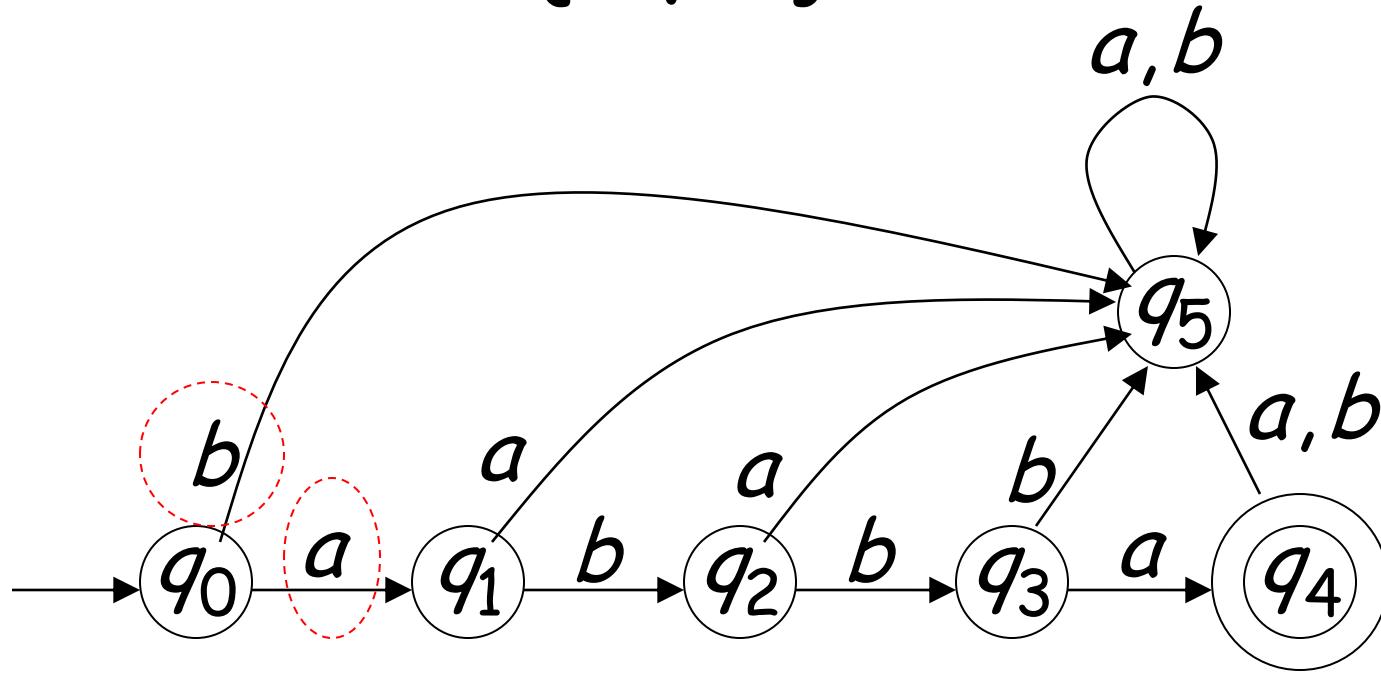
$\delta$	$a$	$b$
$q_0$	$q_1$	$q_5$
$q_1$	$q_5$	$q_2$
$q_2$	$q_5$	$q_3$
$q_3$	$q_4$	$q_5$
$q_4$	$q_5$	$q_5$
$q_5$	$q_5$	$q_5$



# Transition Graph



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



For every state, there is a transition  
for every symbol in the alphabet

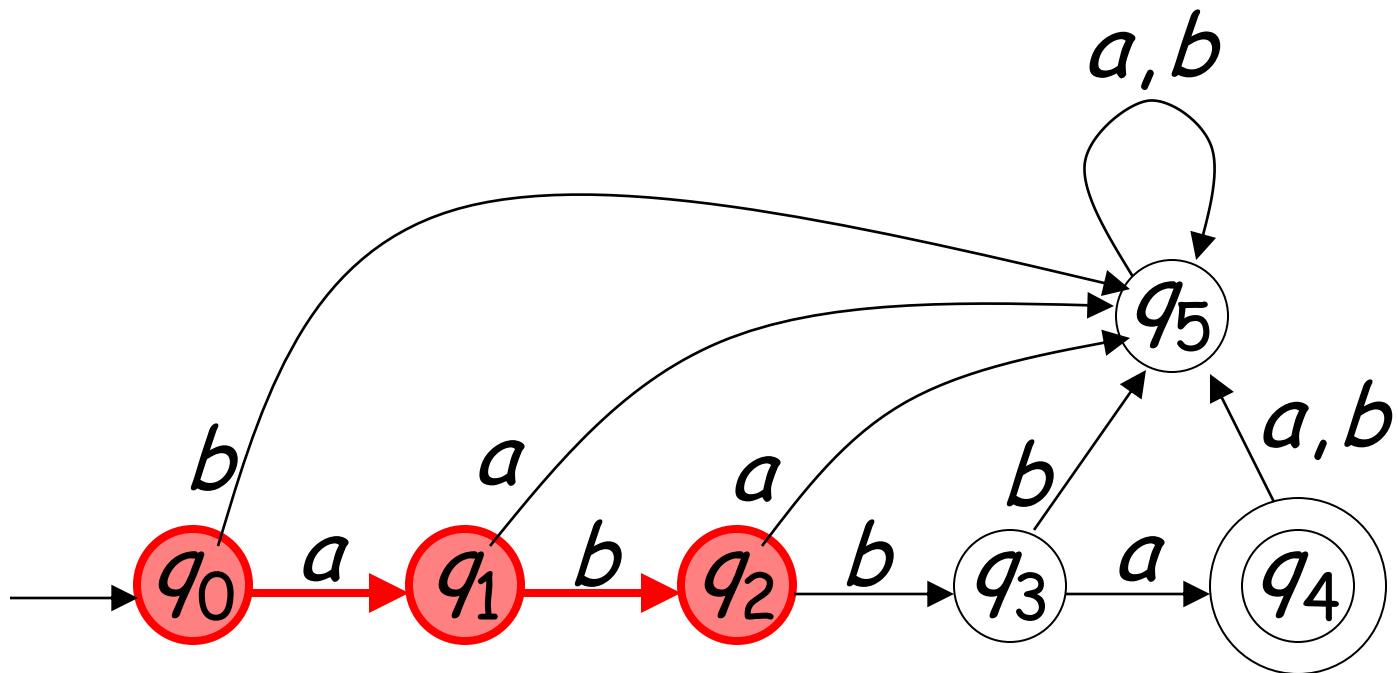
Extended Transition Function

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

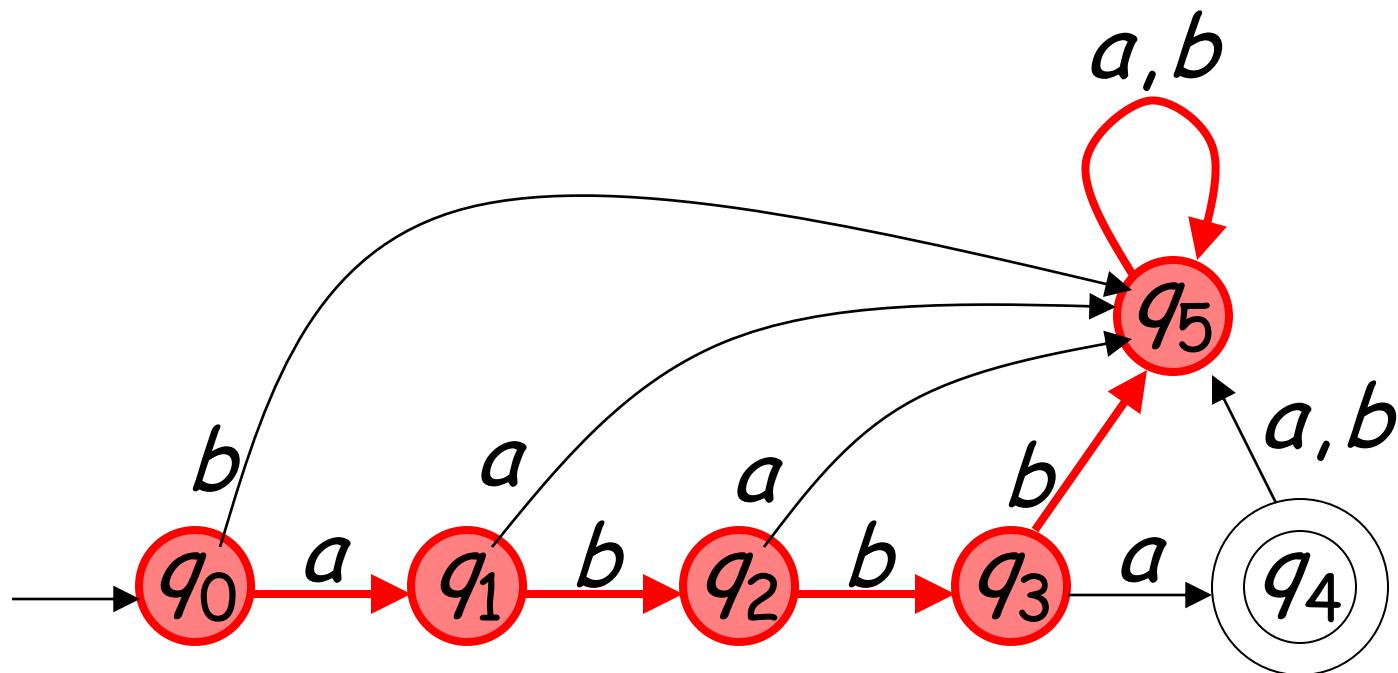
$$\delta^*(q, w) = q'$$

Describes the resulting state  
after scanning string  $w$  from state  $q$

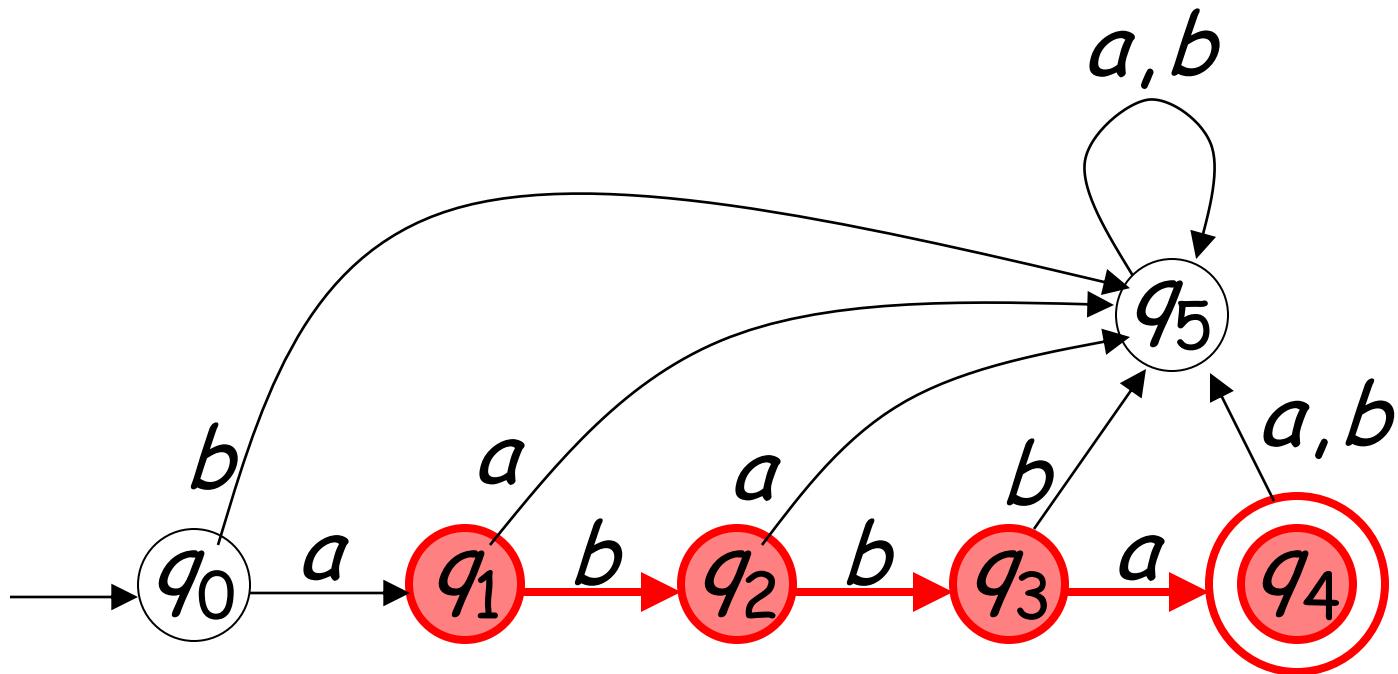
Example:  $\delta^*(q_0, ab) = q_2$



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



$$\delta^*(q_1, bba) = q_4$$



Special case:

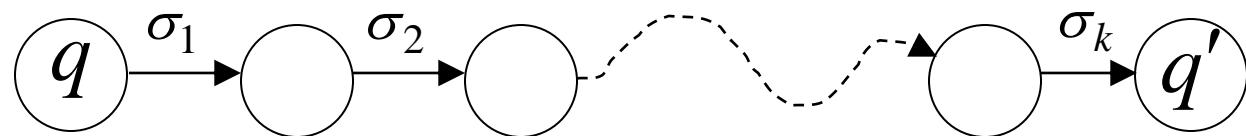
for any state  $q$

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

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

implies that there is a walk of transitions

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



states may be repeated



# Language Accepted by DFA

Language of DFA  $M$  :

it is denoted as  $L(M)$  and contains all the strings accepted by  $M$

We say that a language  $L'$  is accepted (or recognized) by DFA  $M$  if  $L(M) = L'$

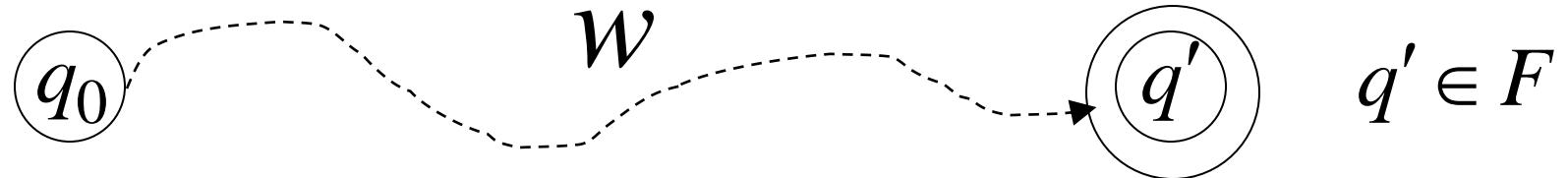
For a DFA

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

Language accepted by

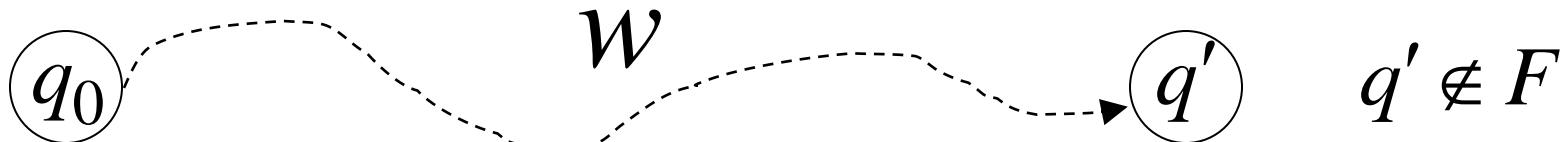
$$M$$

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

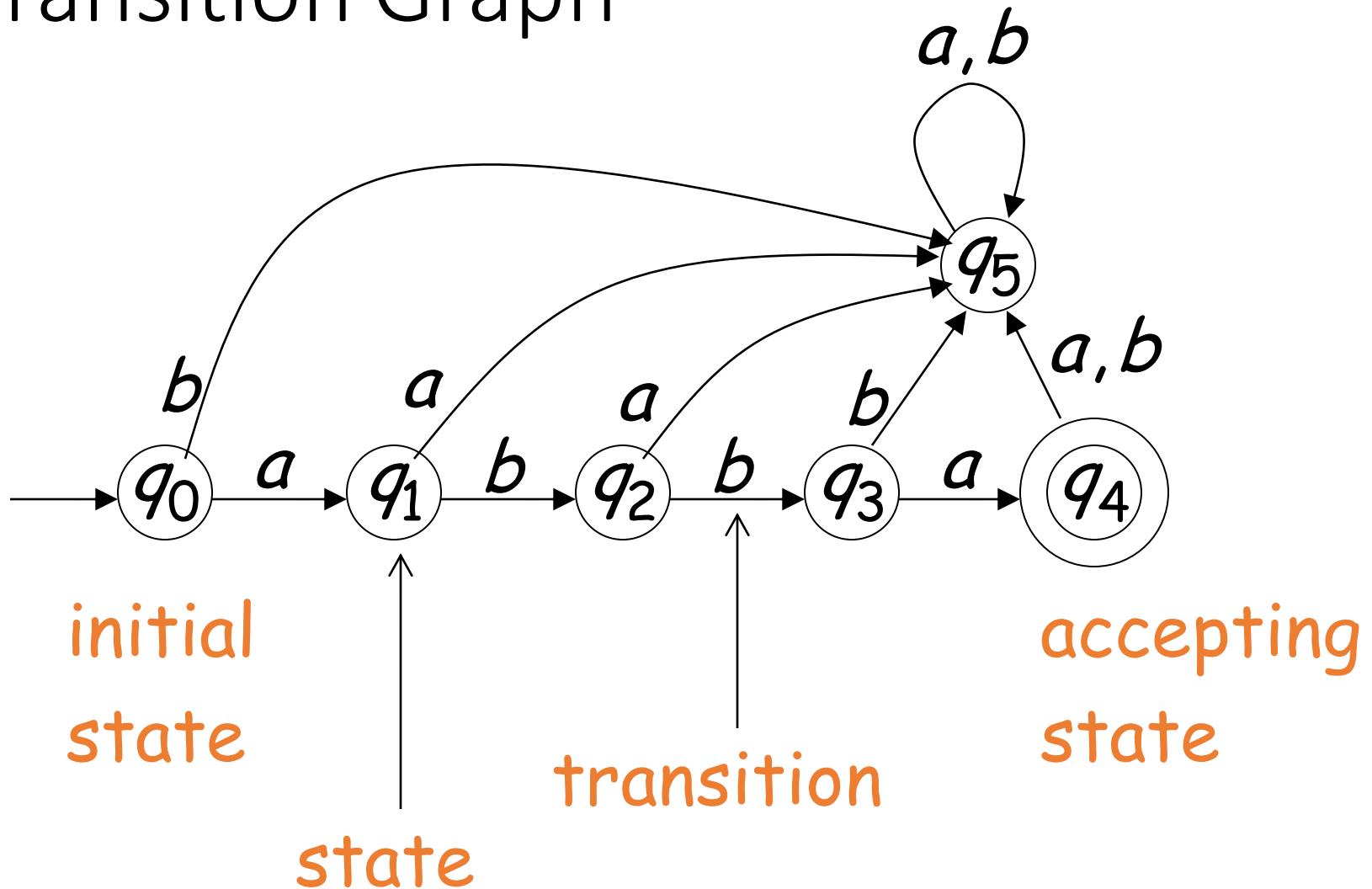


$M$

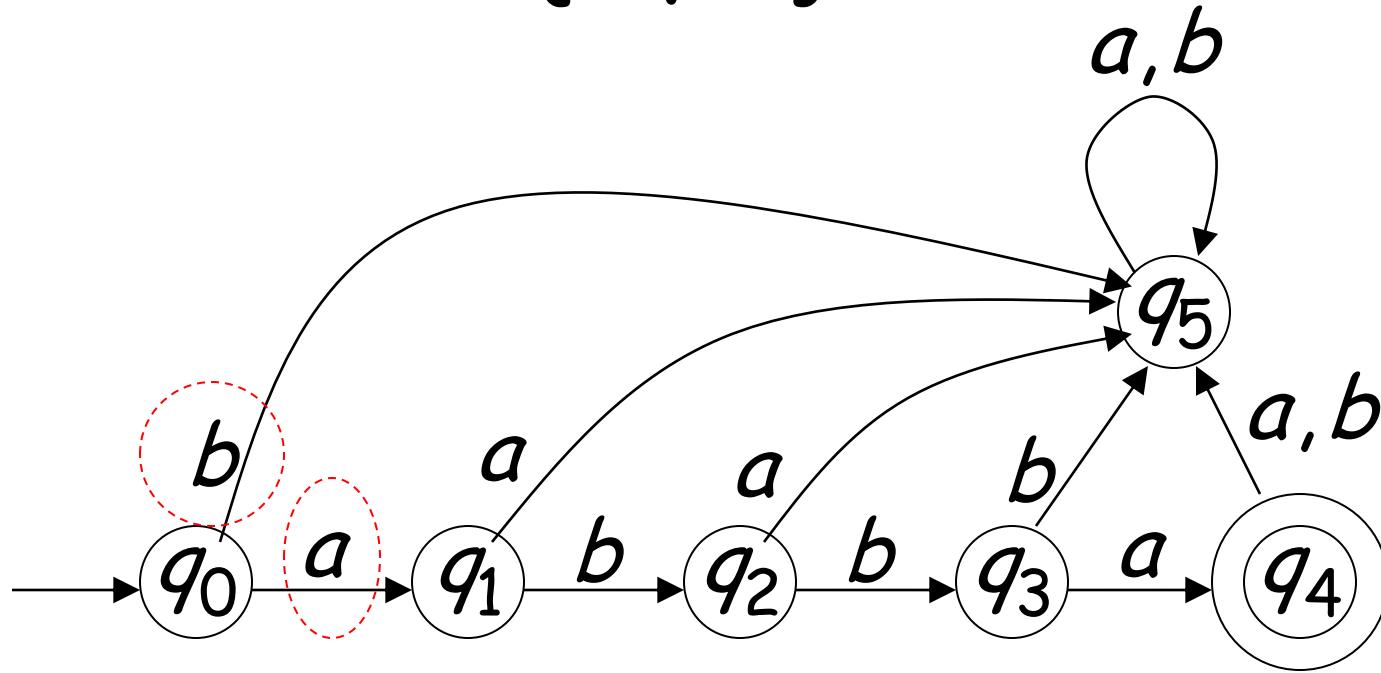
Language rejected by  
 $L(M) = \{w \in \Sigma^* : \delta^*(q_0, w) \notin F\}$



# Transition Graph



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



For every state, there is a transition  
for every symbol in the alphabet

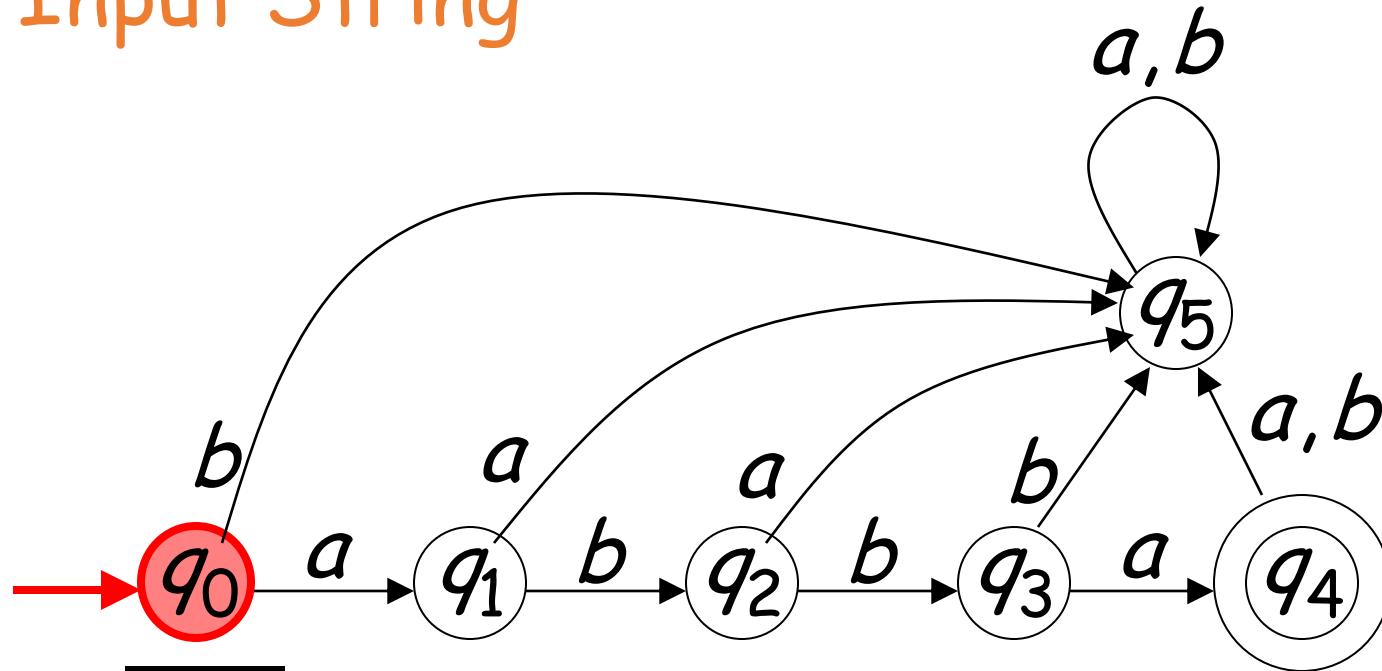
head

Initial Configuration

Input alphabet

a	b	b	a	
---	---	---	---	--

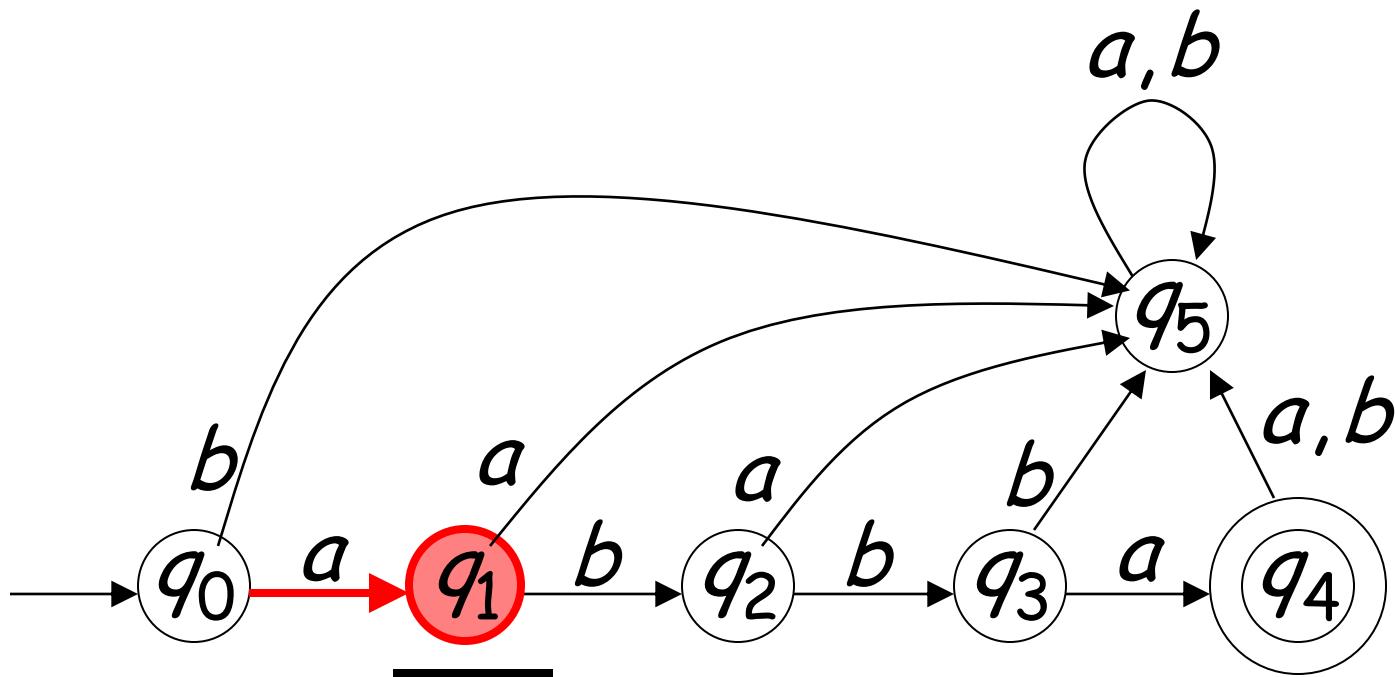
Input String

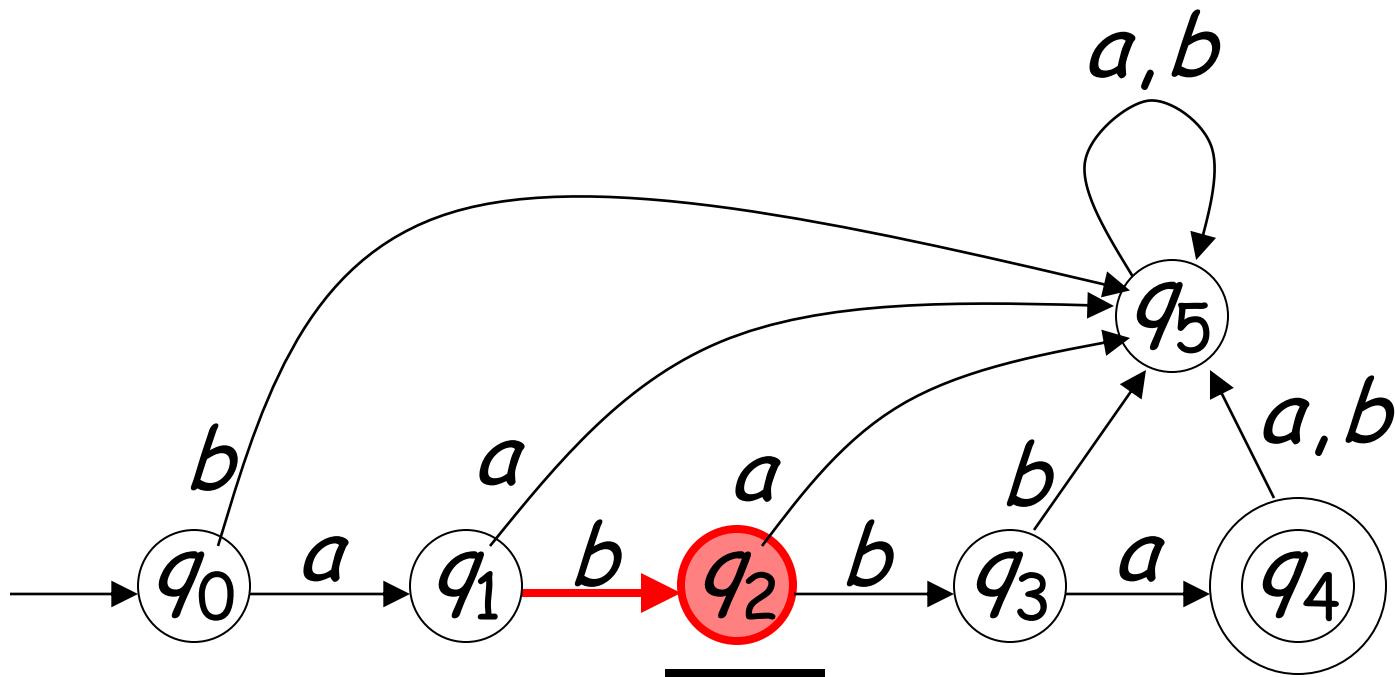
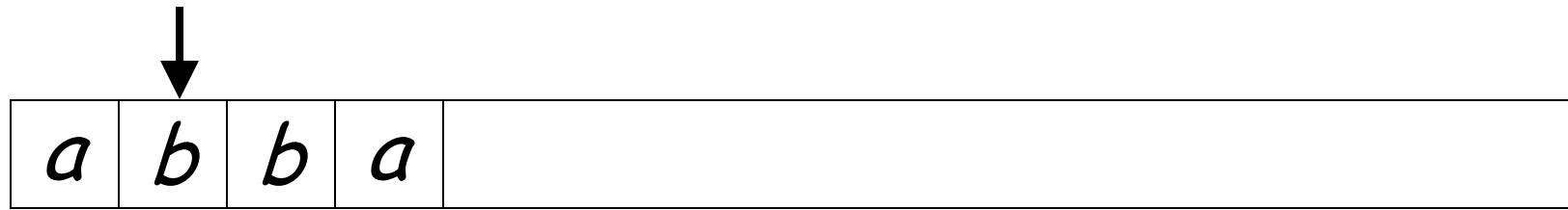


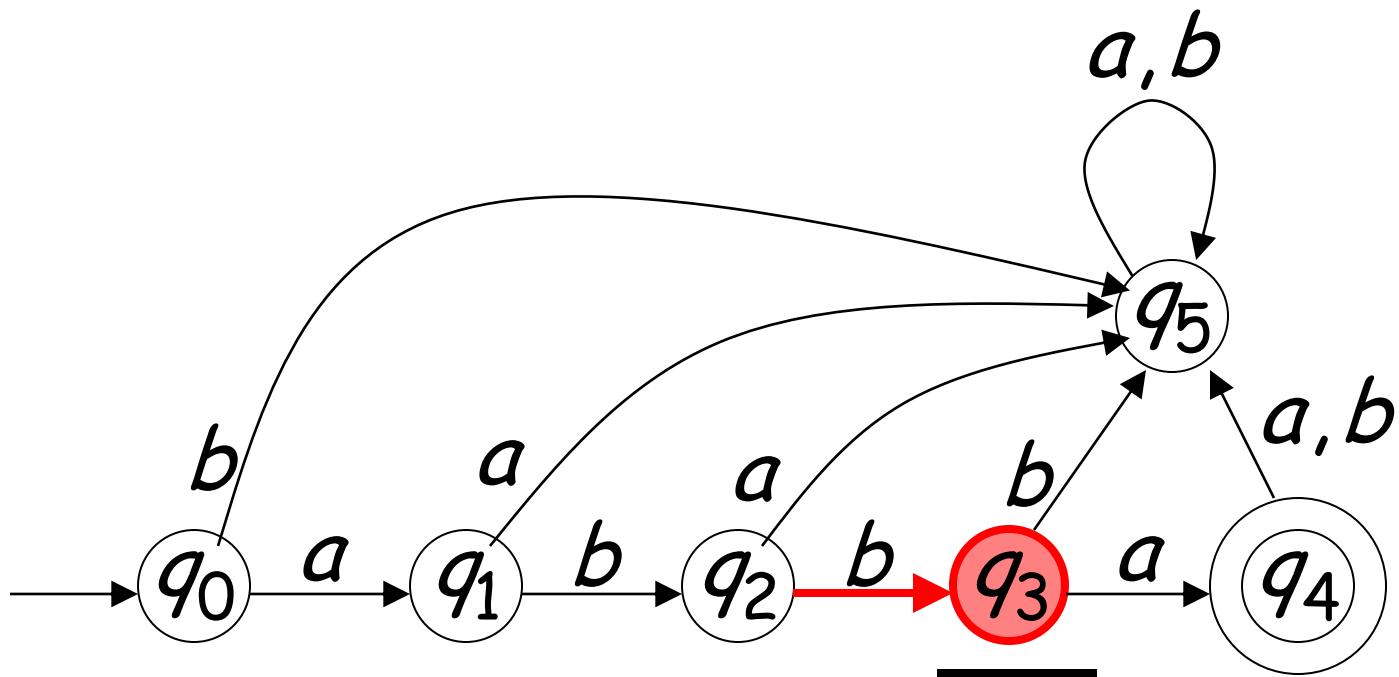
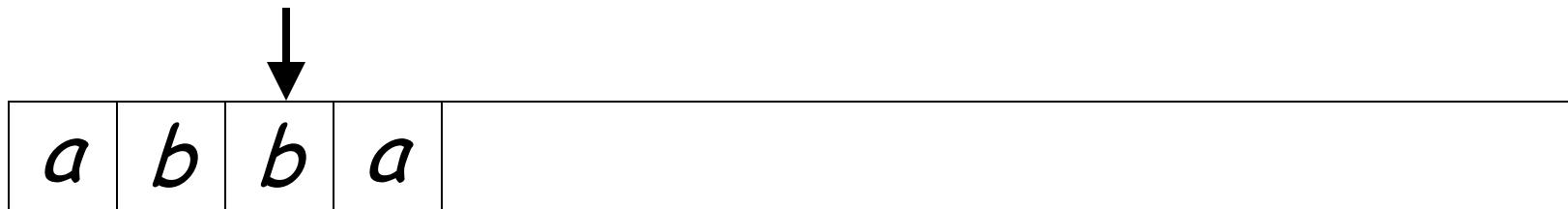
Initial state

# Scanning the Input

a	b	b	a	
---	---	---	---	--



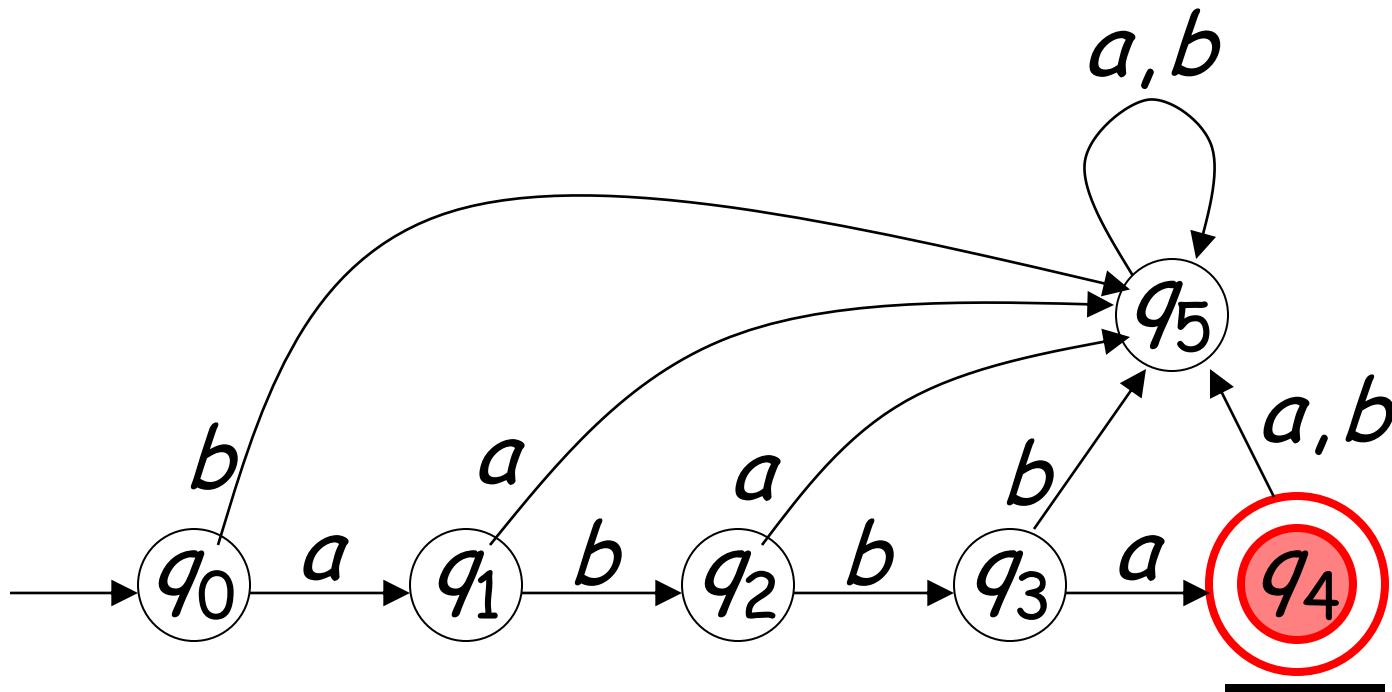




# Input finished



a	b	b	a	
---	---	---	---	--



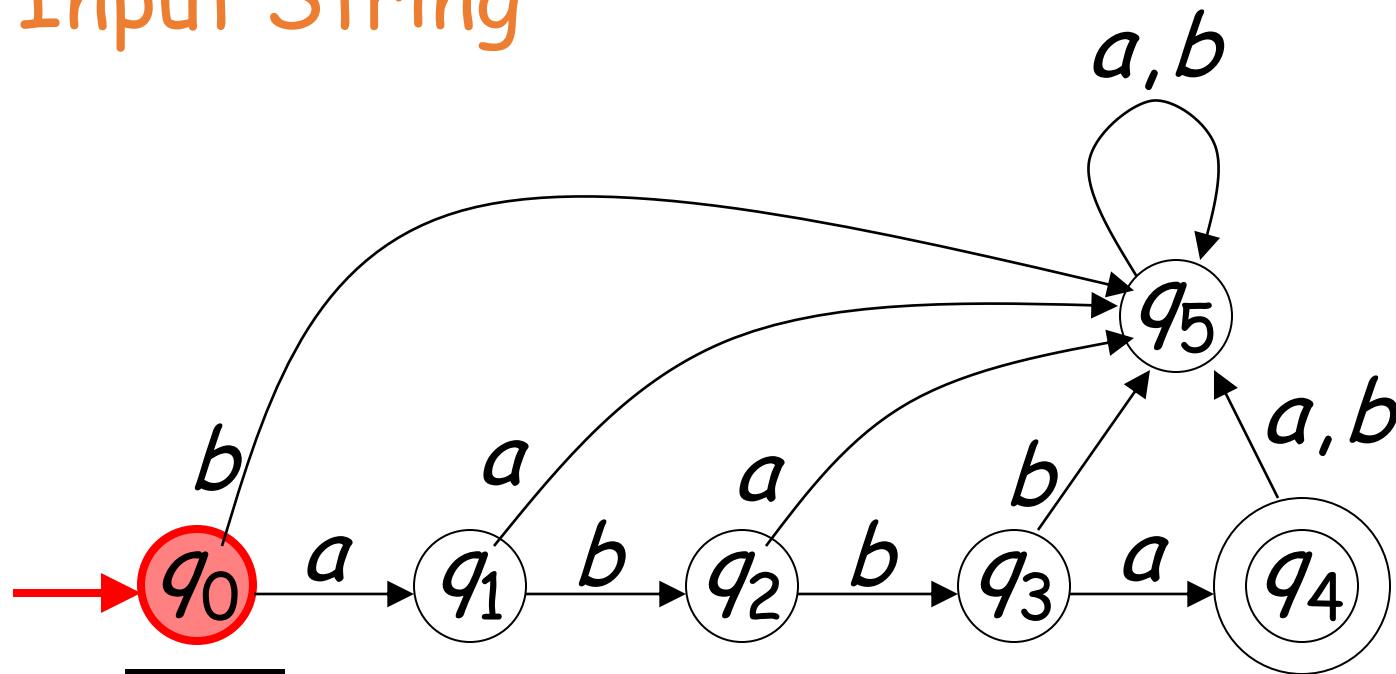
accept

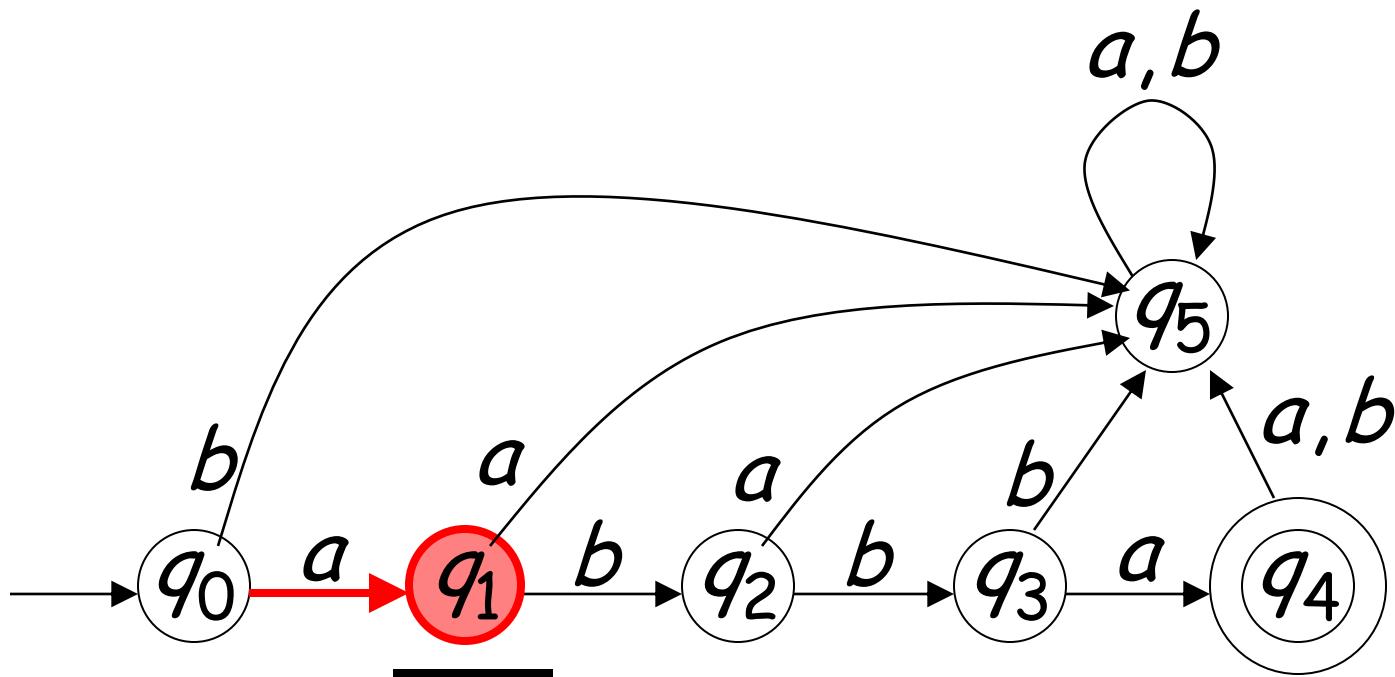
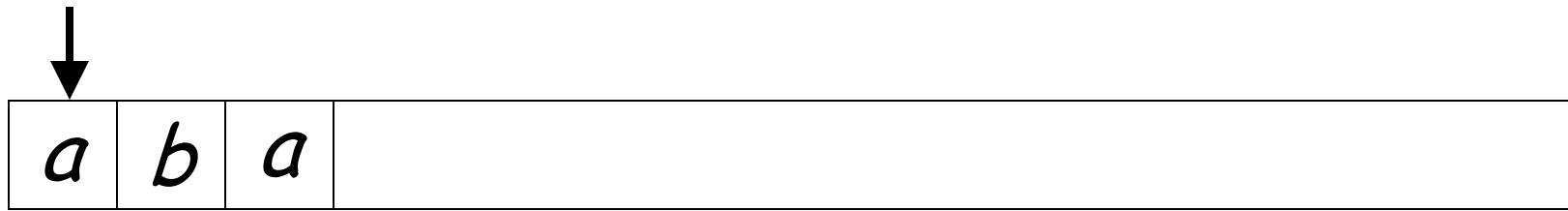
# A Rejection Case



a	b	a	
---	---	---	--

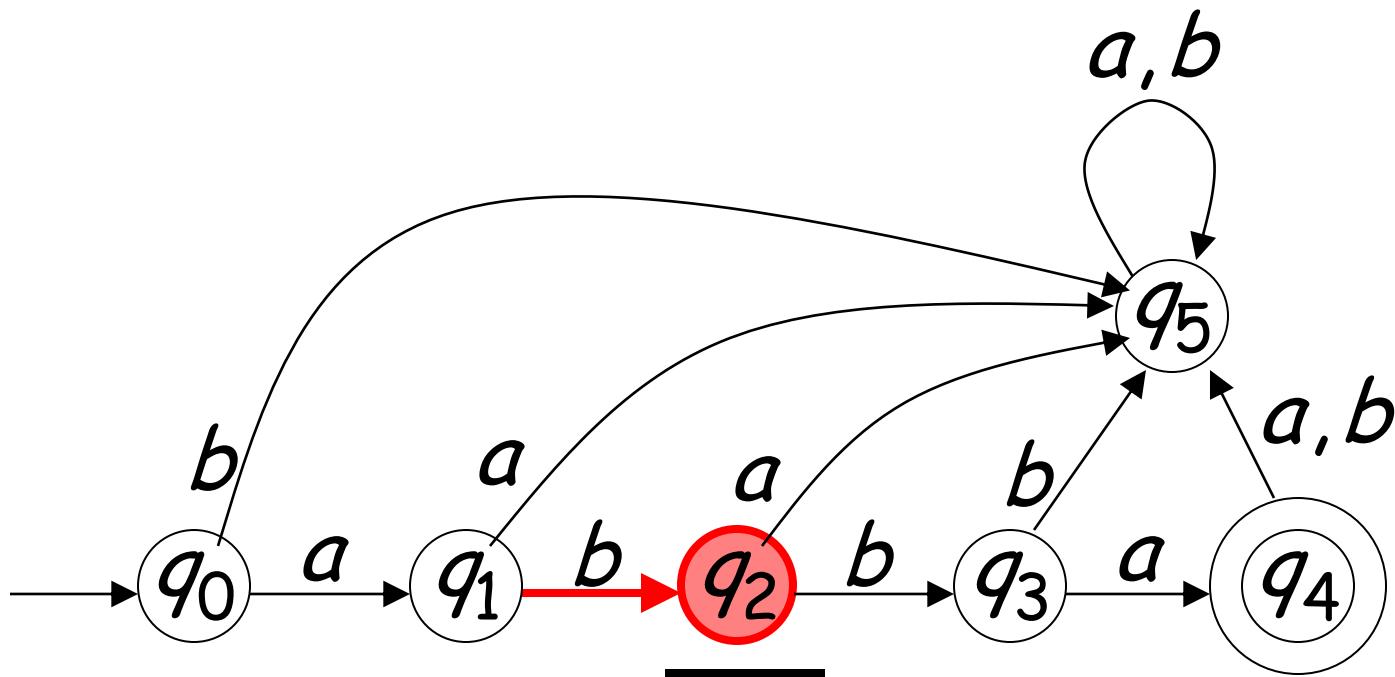
Input String





↓

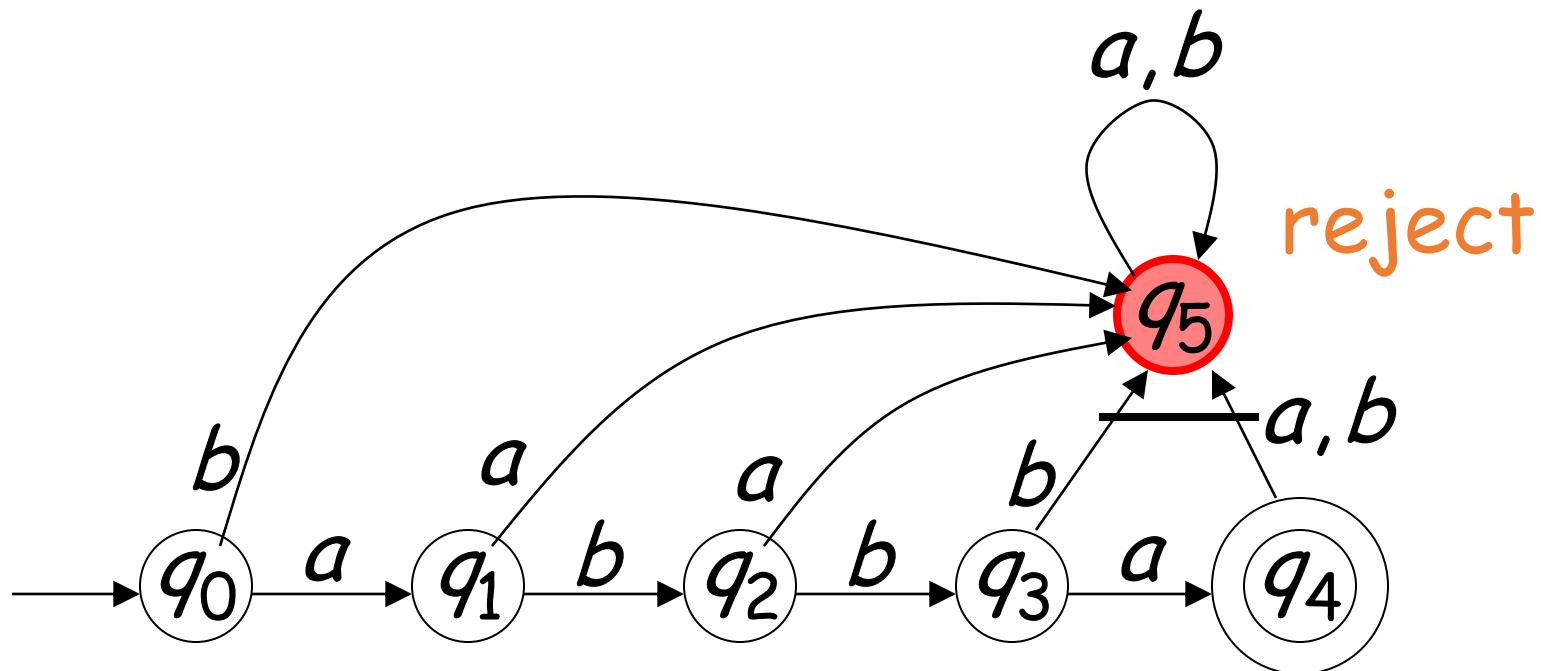
$a$	$b$	$a$	
-----	-----	-----	--



Input finished



a	b	a	
---	---	---	--

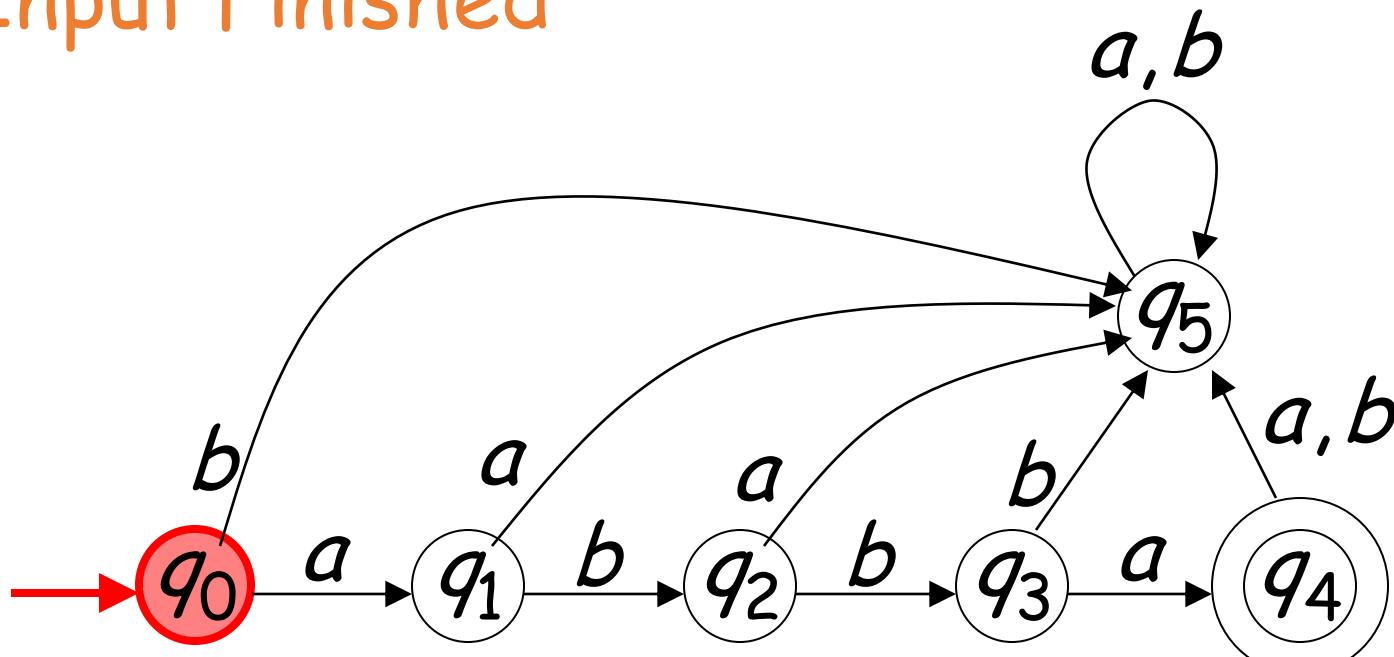


# Another Rejection Case

Input String is empty

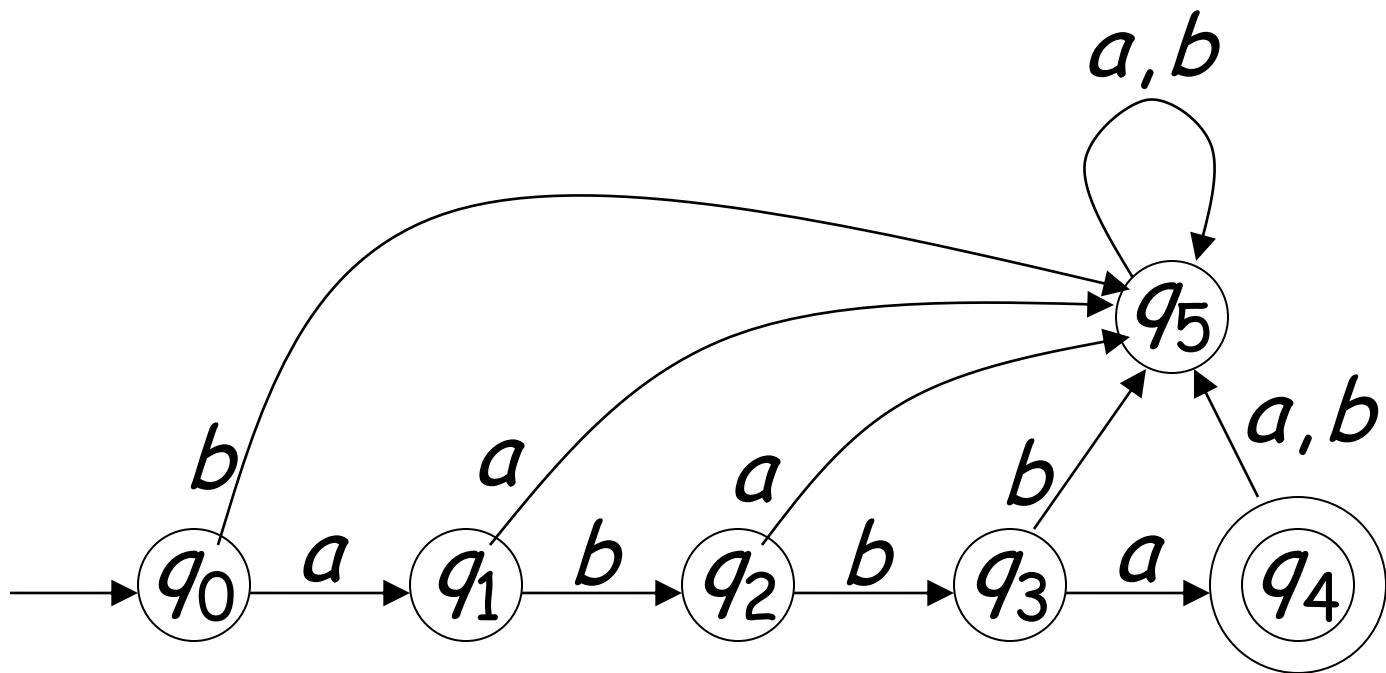
$(\lambda)$

Input Finished



reject

Language Accepted:  $L = \{abba\}$



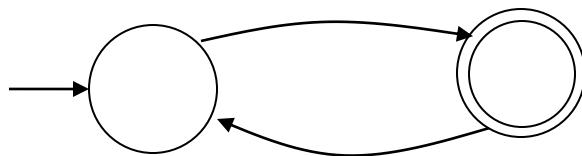
To accept a string:

all the input string is scanned  
and the last state is accepting

To reject a string:

all the input string is scanned  
and the last state is non-accepting

# Example of a finite automaton



- There are **states** off and on, the automaton **starts** in off and tries to reach the “**good state**” on
- What sequences of  $f$ s lead to the good state?
- Answer:  $\{f, fff, fffff, \dots\} = \{f^n : n \text{ is odd}\}$
- This is an **example** of a deterministic finite automaton over alphabet  $\{f\}$

---

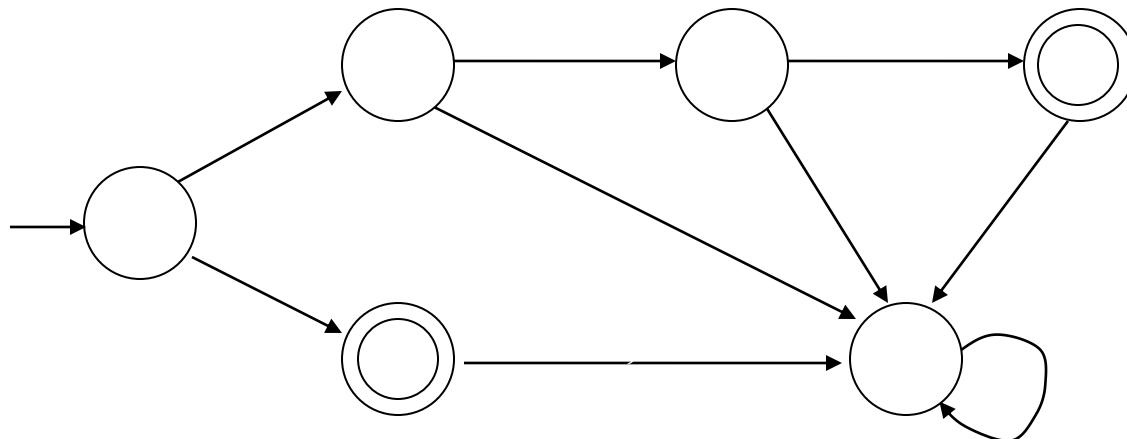
# Examples

- Construct a DFA that accepts the language

# Examples

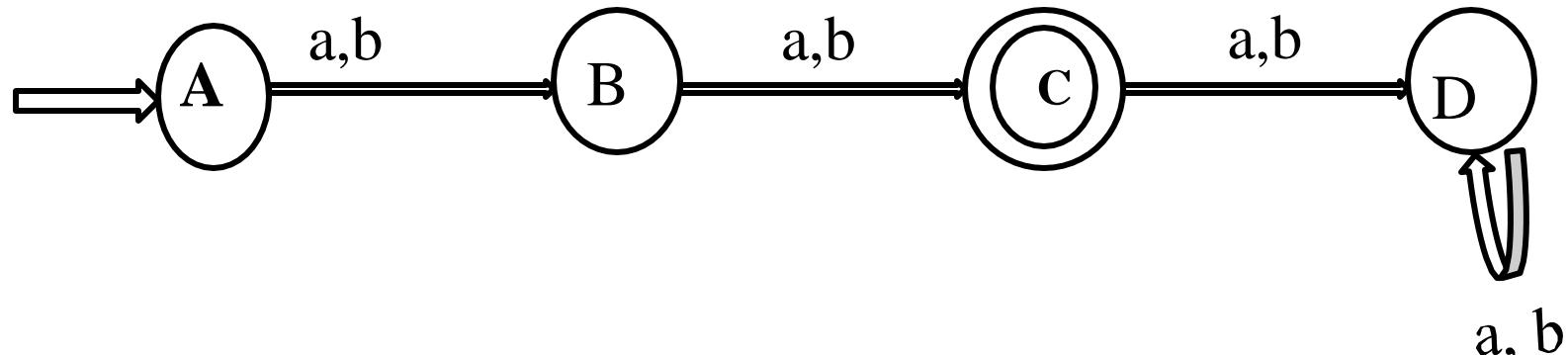
- Construct a DFA that accepts the language

- Answer



## Example

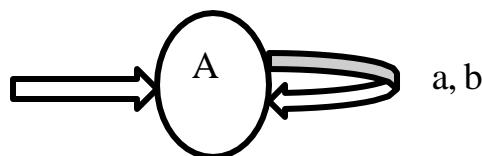
Construct a DFA which accepts set of all string over  $\{a, b\}$  of length 2.  $L = \{aa, ab, ba, bb\}$



String Accept:-scan the entire string, if we reach a final state from initial state.

language Accept:-All the string in language are “accepted” and all string which are not in the language are “rejected”

## Example

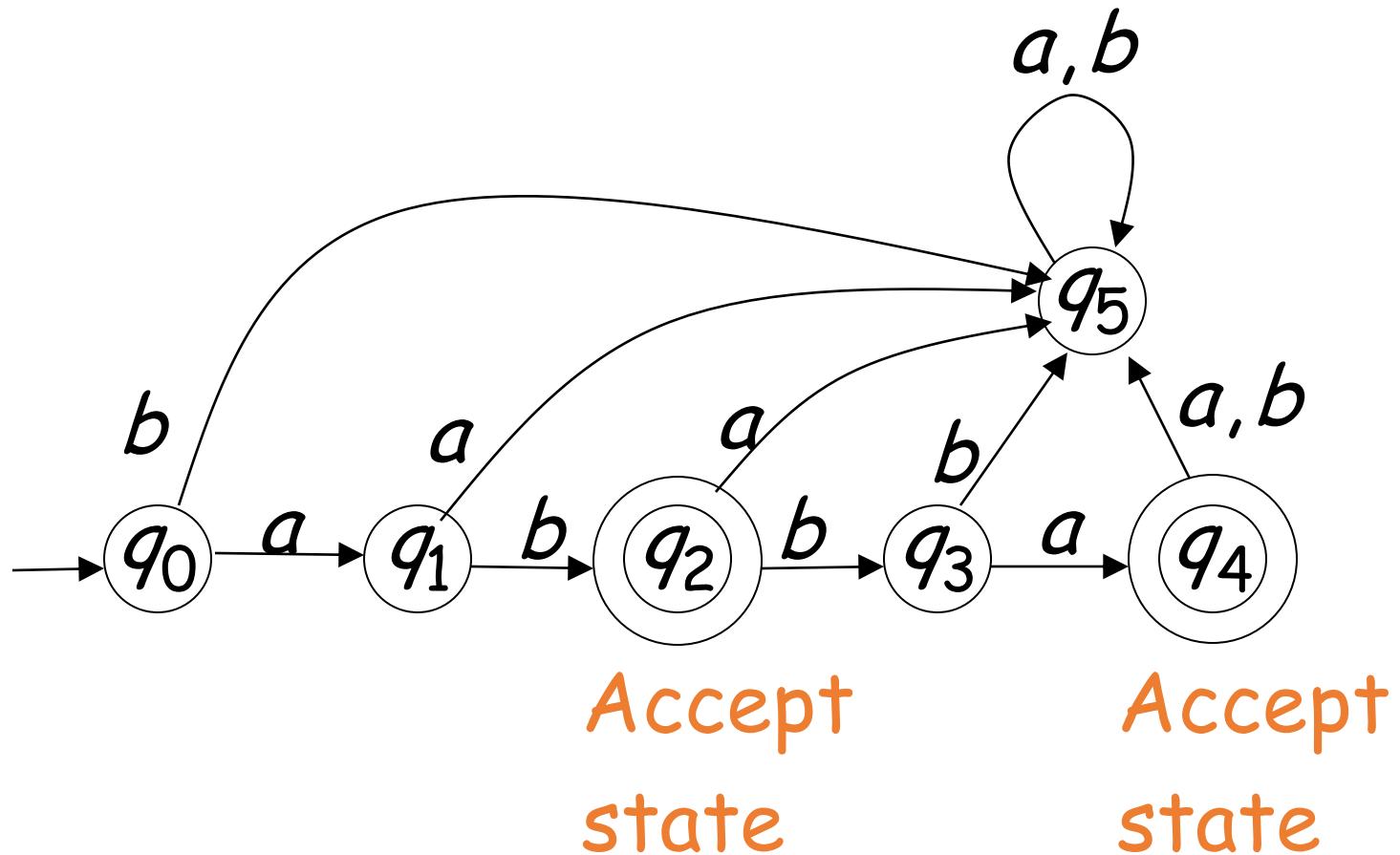


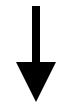
Above DFA is incorrect because it also accept the string which is not in language



# Another Example

$$L = \{ab, abba\}$$

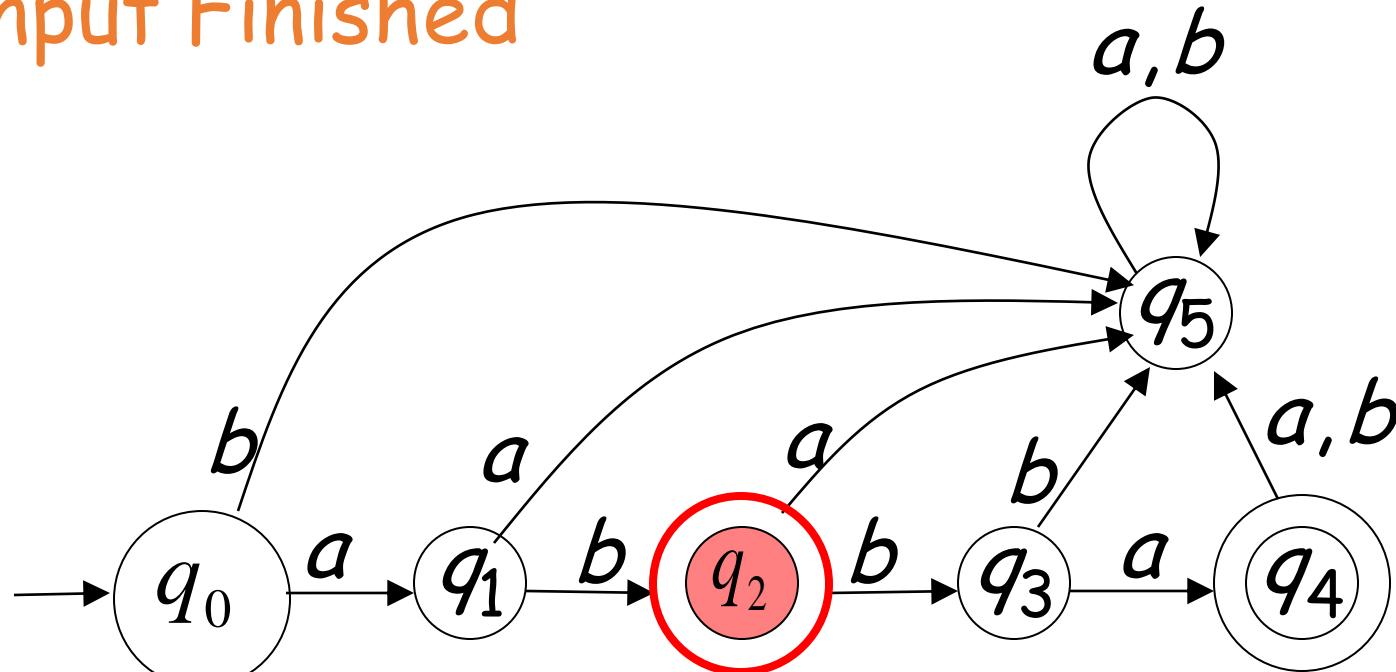




Input Tape

$ab$

Input Finished



accept

# Language Accepted by DFA

Language of DFA  $M$ :

it is denoted as  $L(M)$  and contains all the strings accepted by  $M$

We say that a language  $L'$  is accepted (or recognized) by DFA  $M$  if  $L(M) = L'$

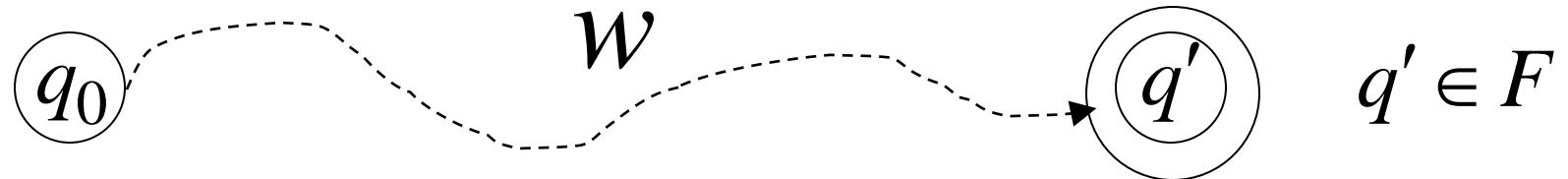
For a DFA

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

Language accepted by

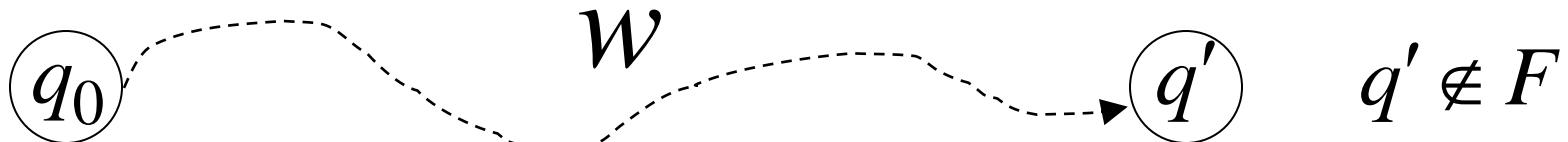
$$M$$

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

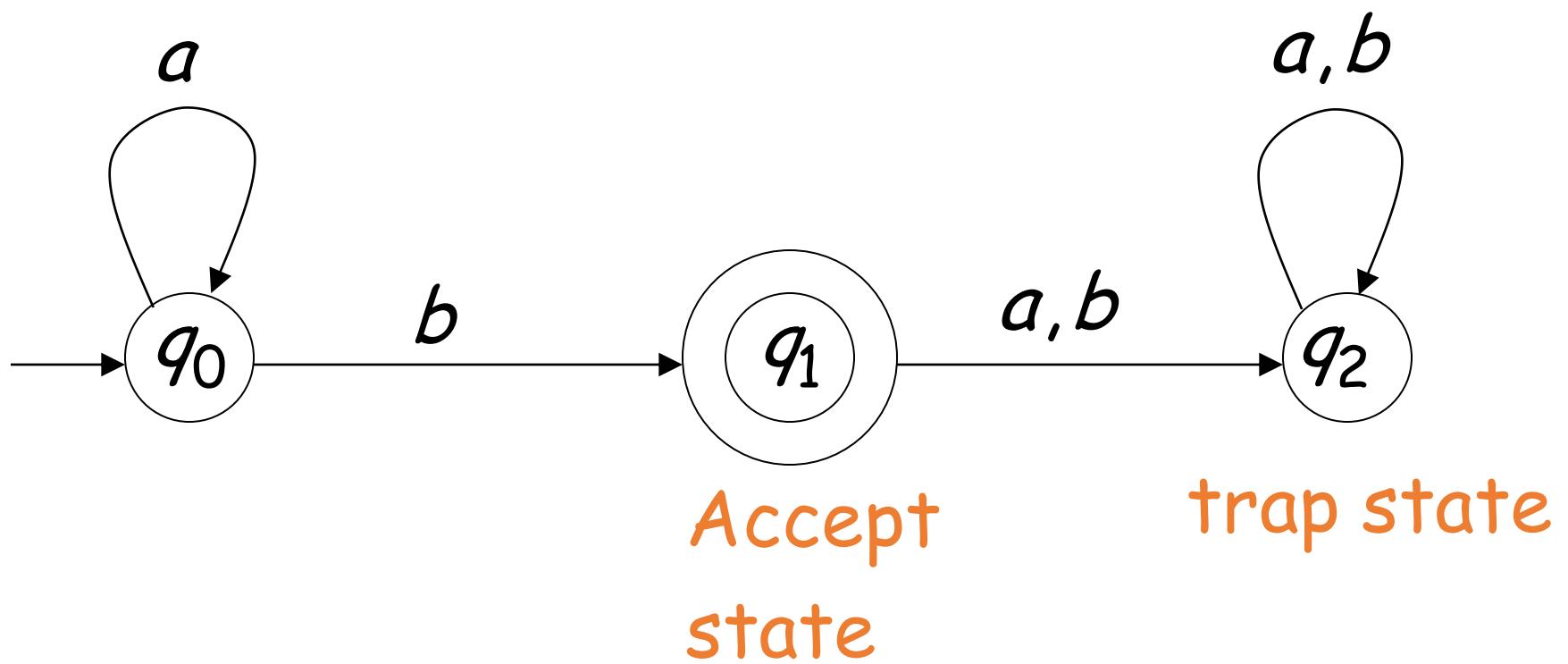


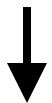
$M$

Language rejected by  
 $L(M) = \{w \in \Sigma^* : \delta^*(q_0, w) \notin F\}$



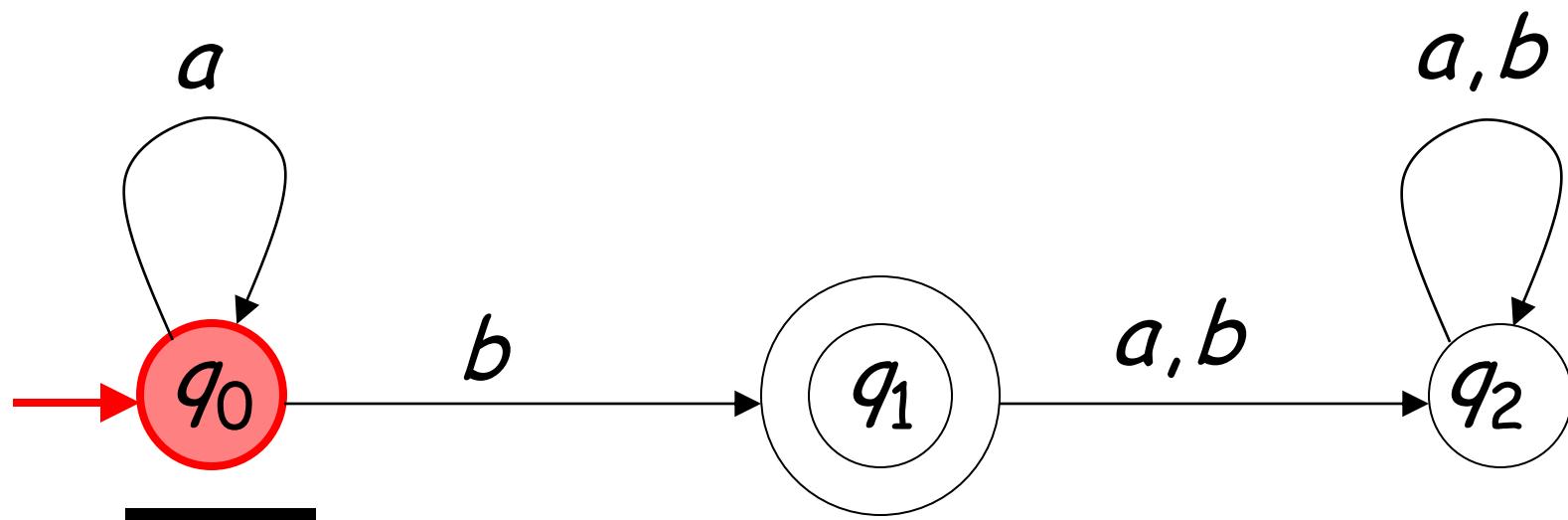
# Another Example





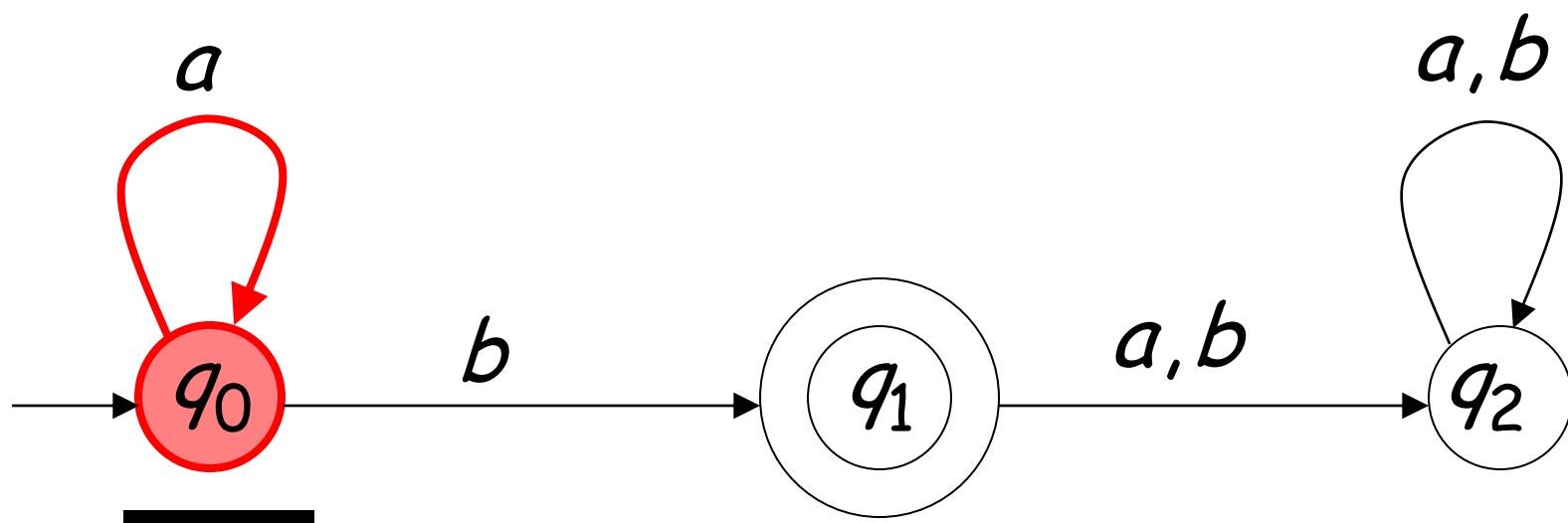
$a$	$a$	$b$	
-----	-----	-----	--

Input String



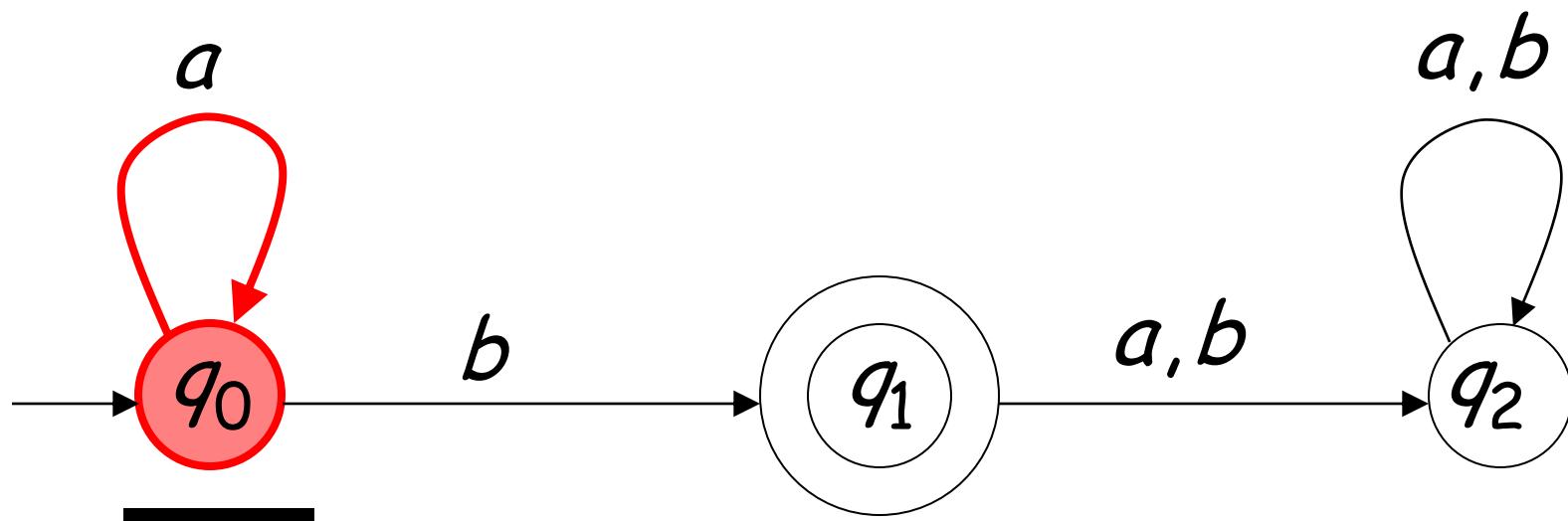
↓

$a$	$a$	$b$	
-----	-----	-----	--

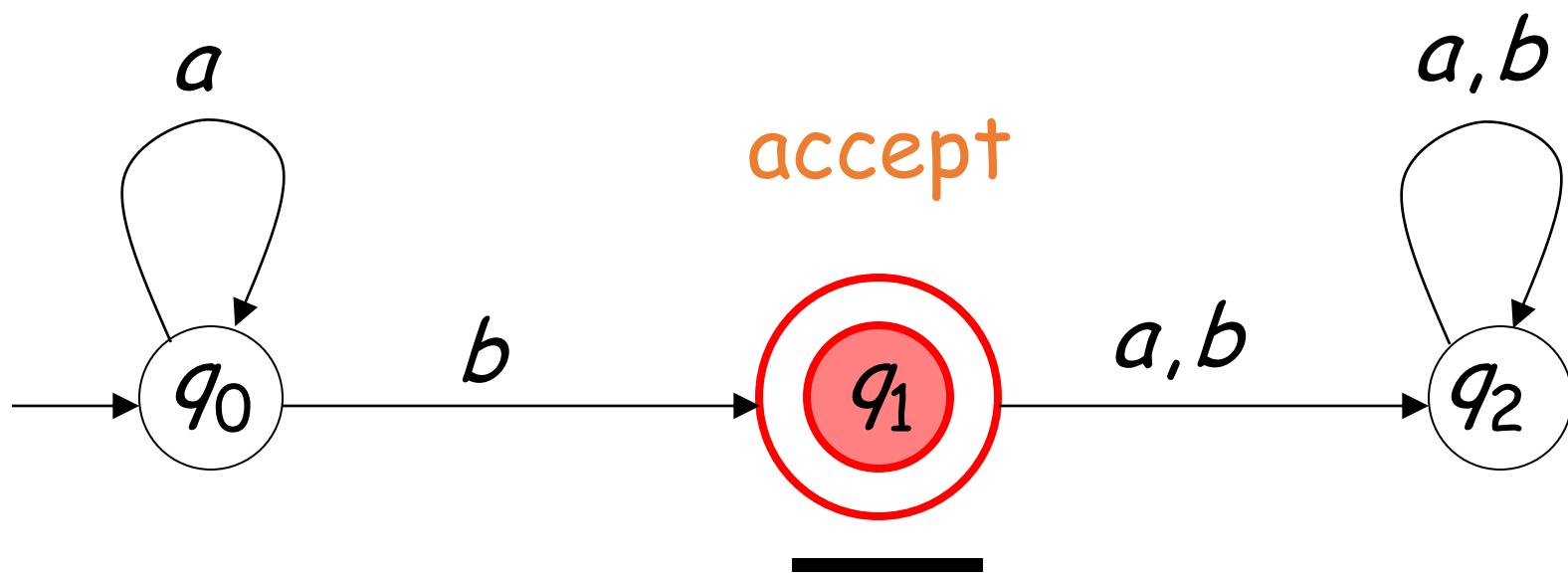
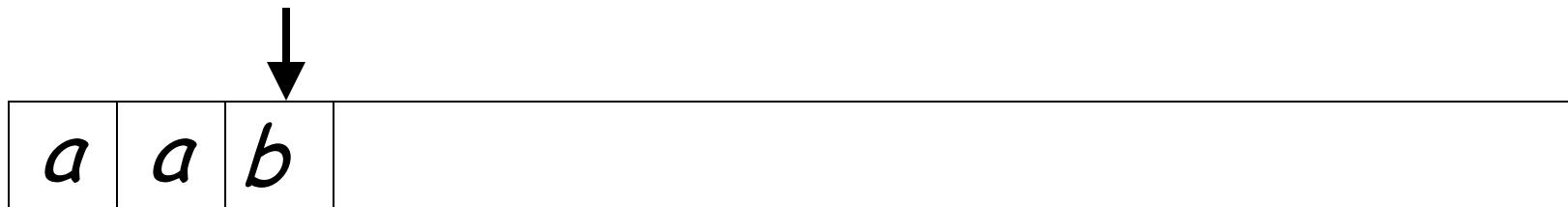


↓

$a$	$a$	$b$	
-----	-----	-----	--



# Input finished

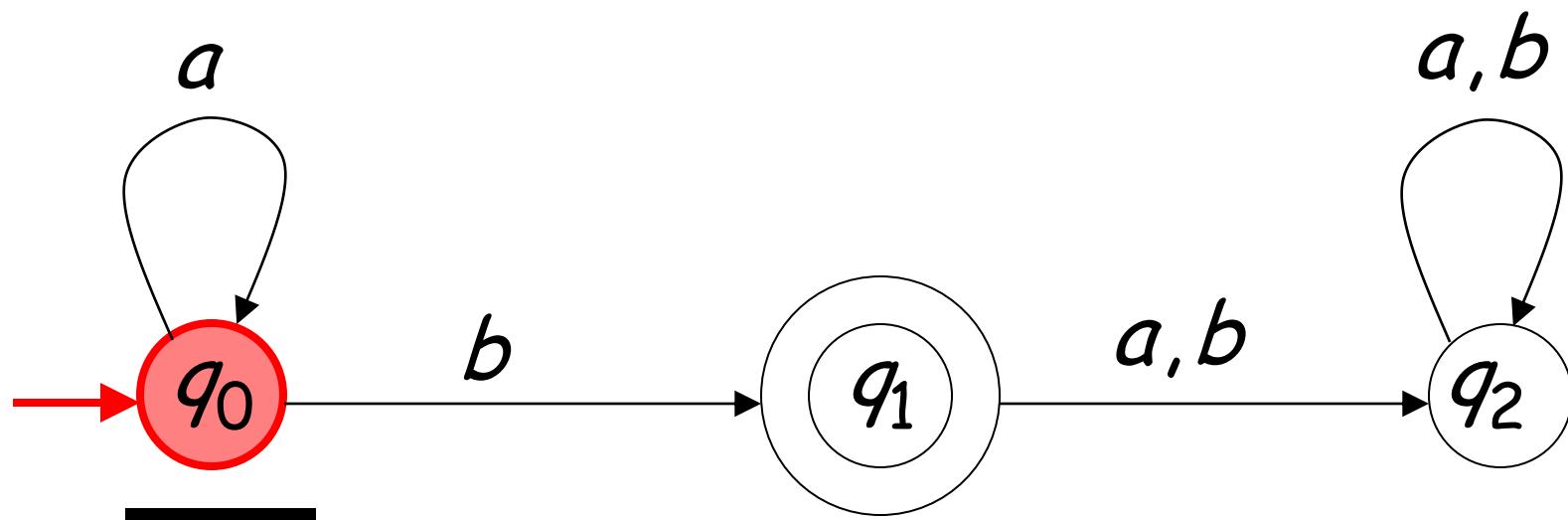


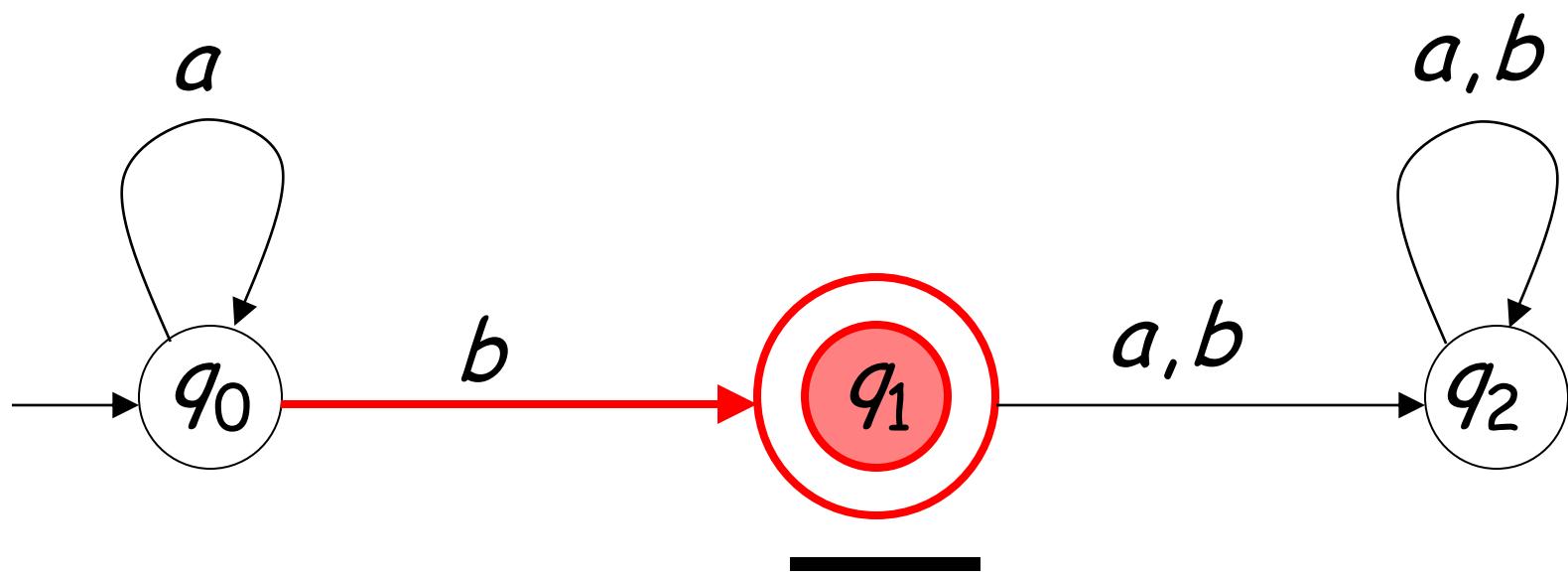
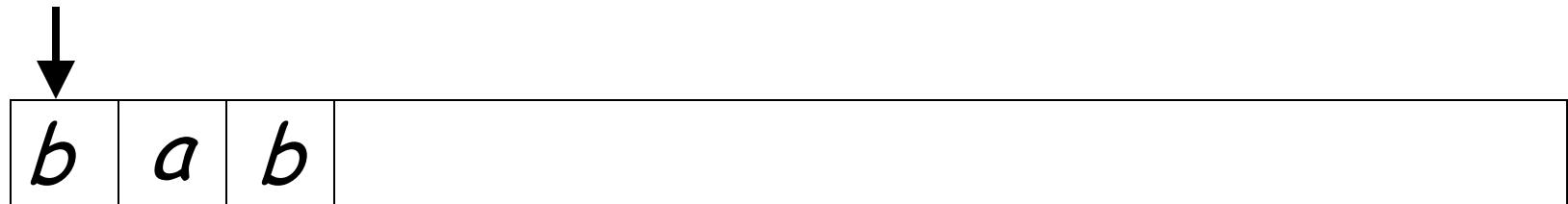
# A rejection case

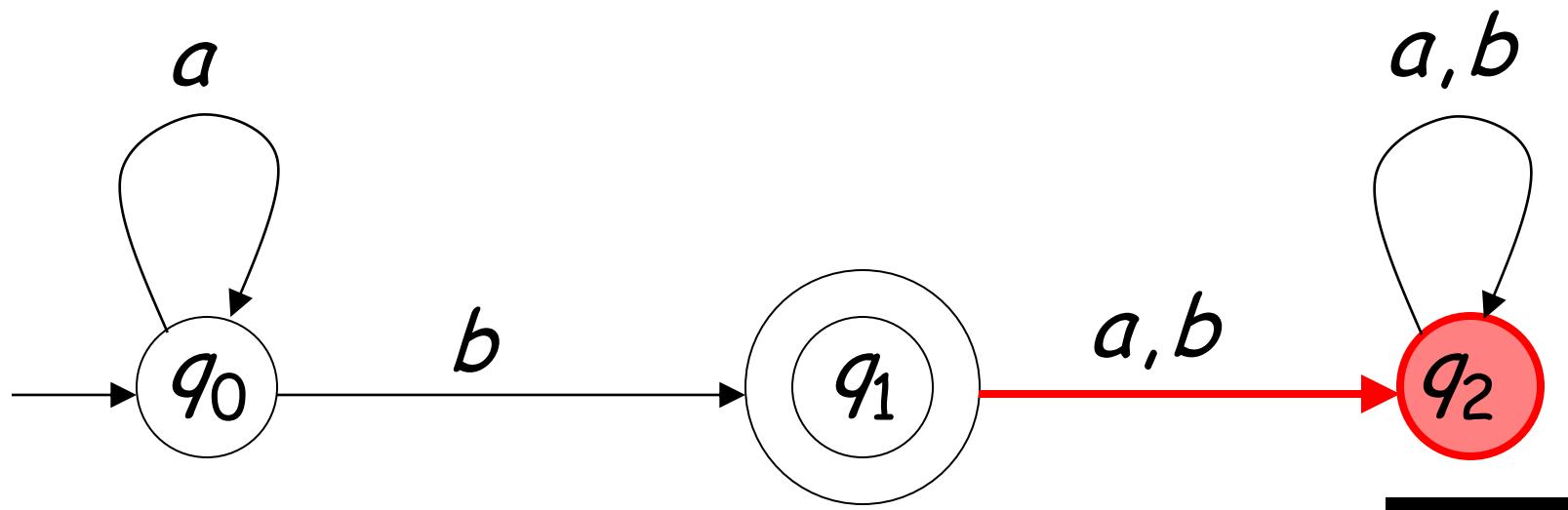
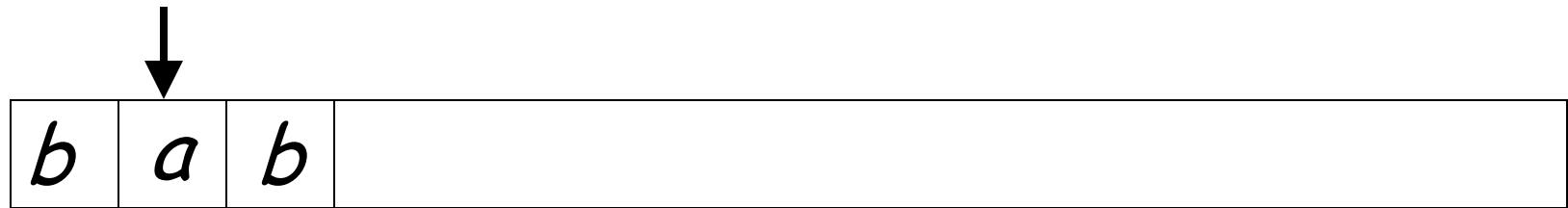


b	a	b	
---	---	---	--

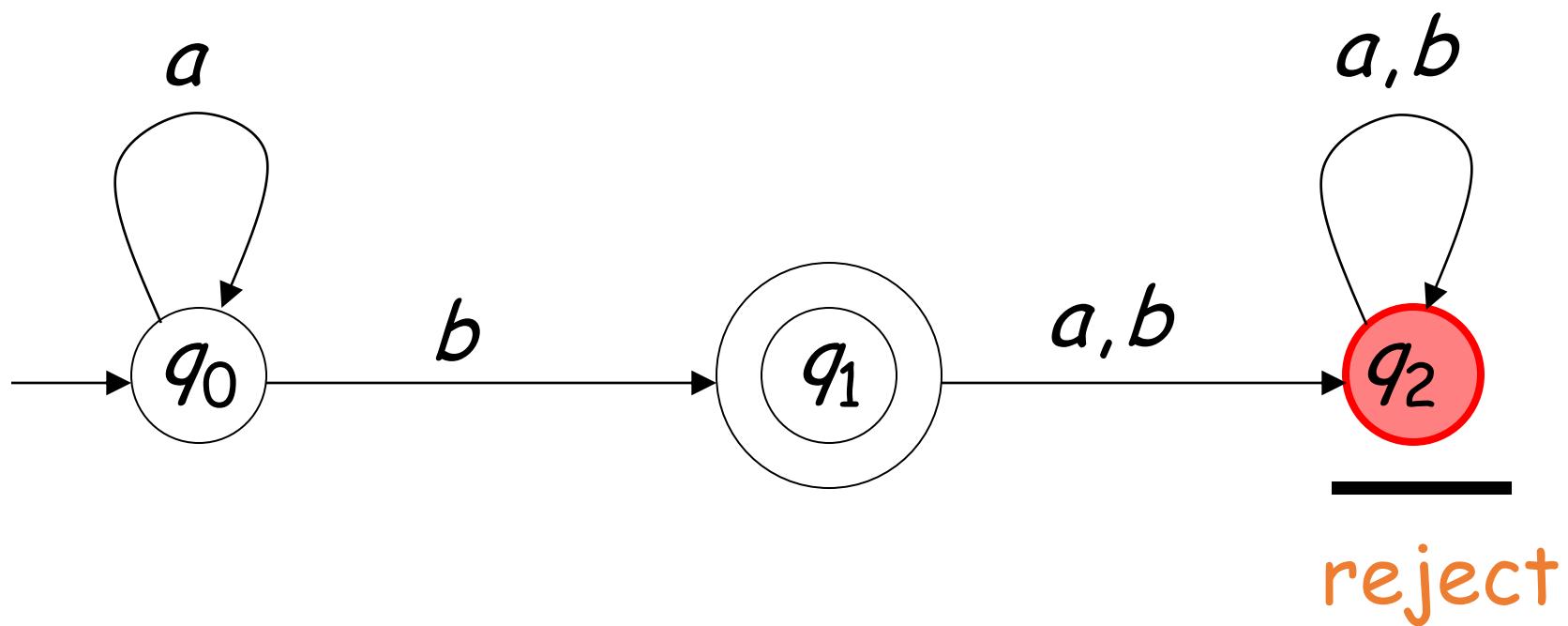
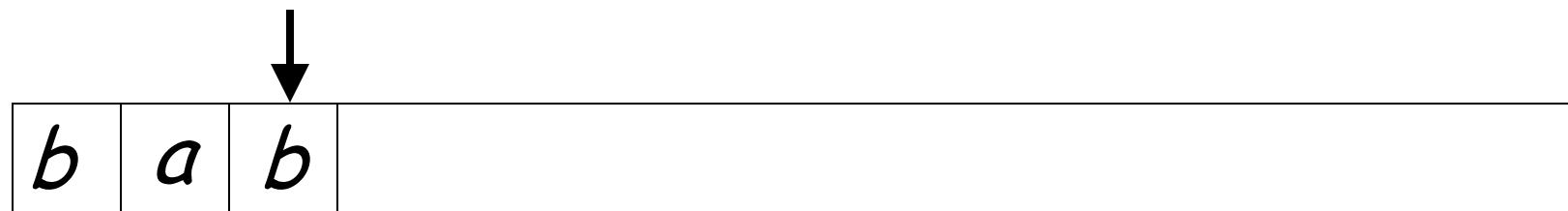
Input String



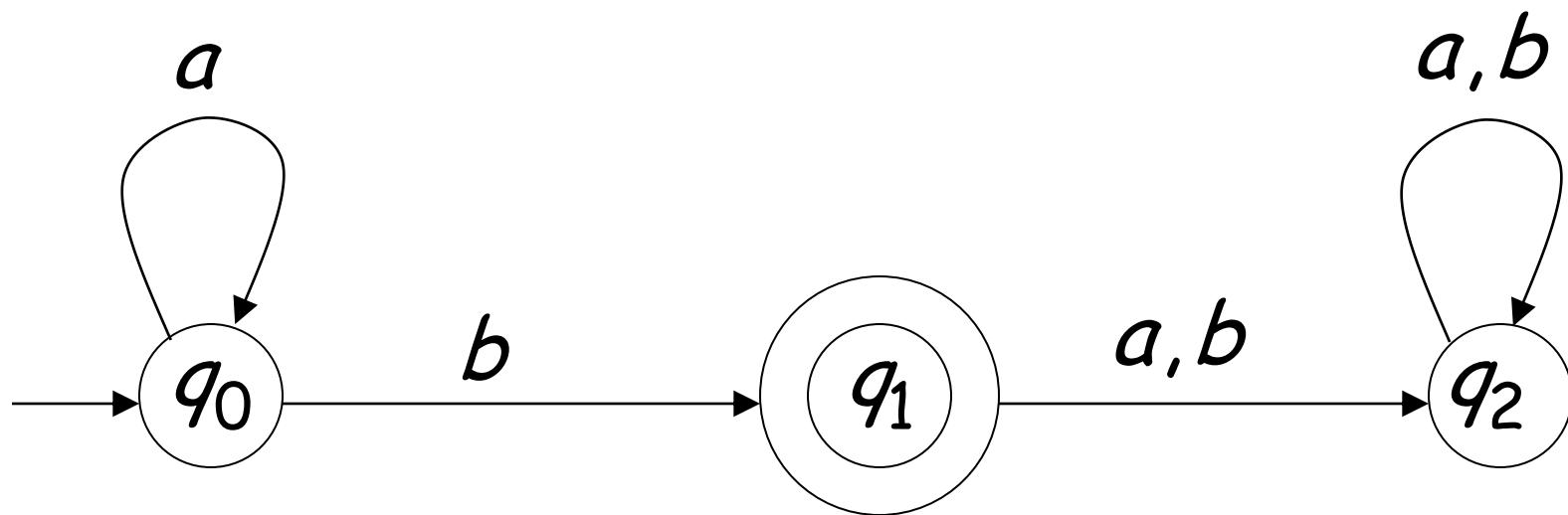




Input finished

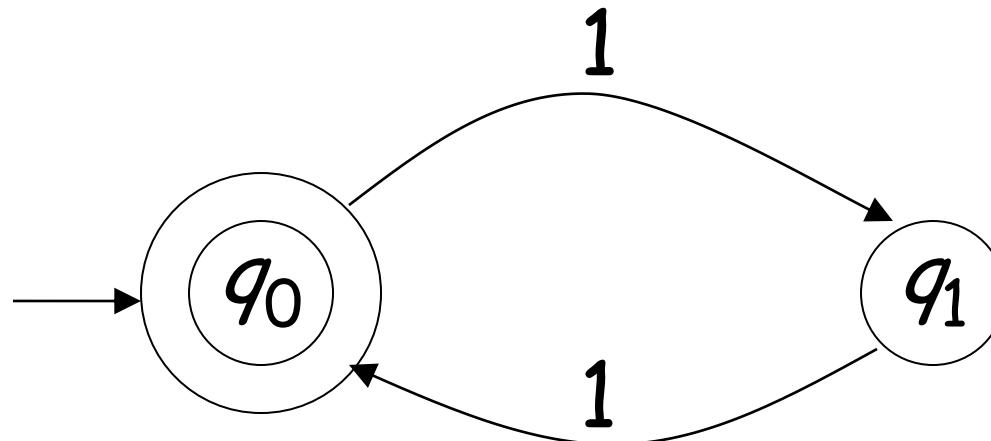


Language Accepted:  $L = \{a^n b : n \geq 0\}$



# Another Example

Alphabet:  $\Sigma = \{1\}$

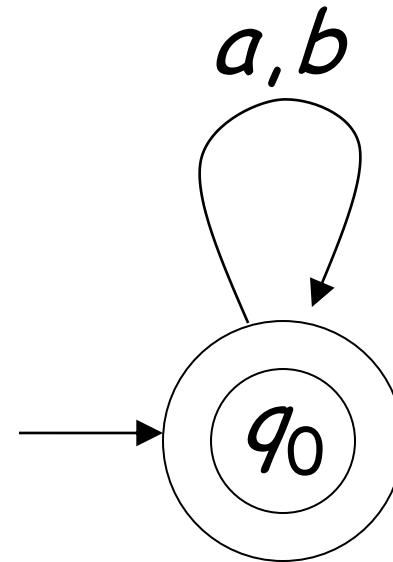
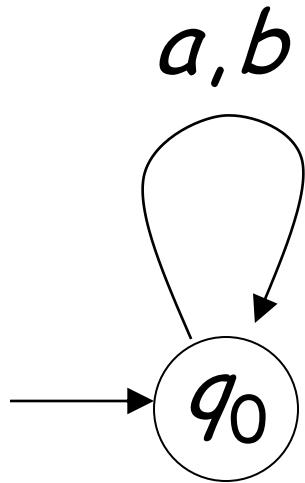


Language Accepted:

$$\begin{aligned} EVEN &= \{x : x \in \Sigma^* \text{ and } x \text{ is even}\} \\ &= \{\lambda, 11, 1111, 111111, \dots\} \end{aligned}$$

# More DFA Examples

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



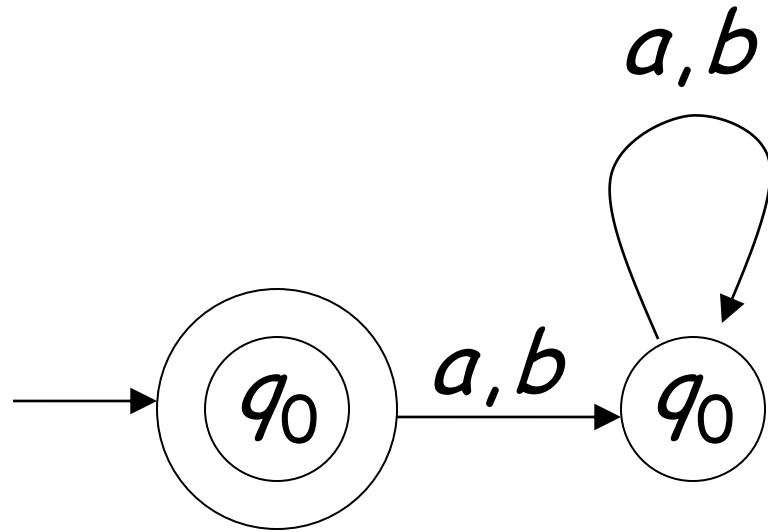
$$L(M) = \{ \}$$

$$L(M) = \Sigma^*$$

Empty language

All strings

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

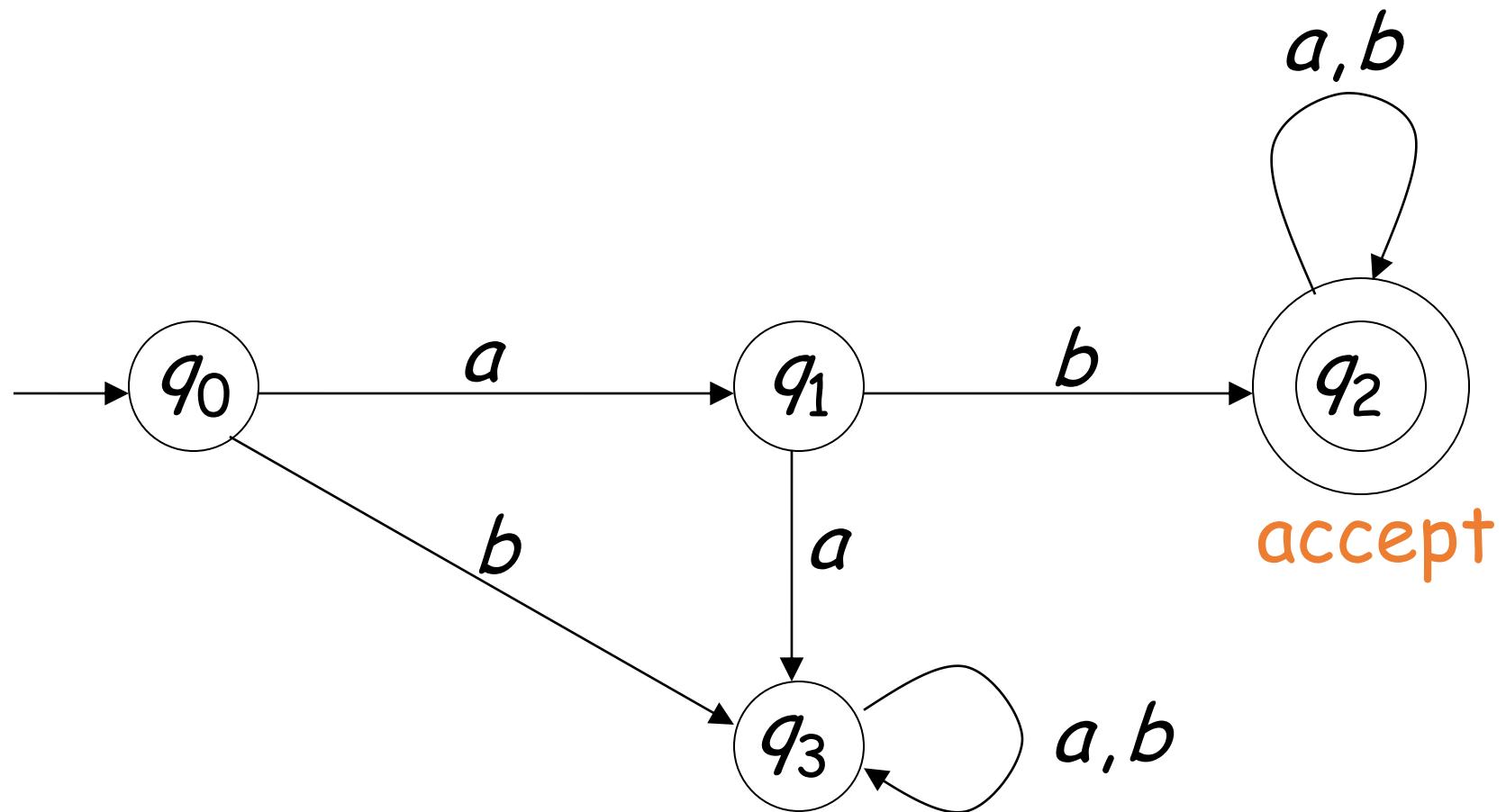


$$L(M) = \{\lambda\}$$

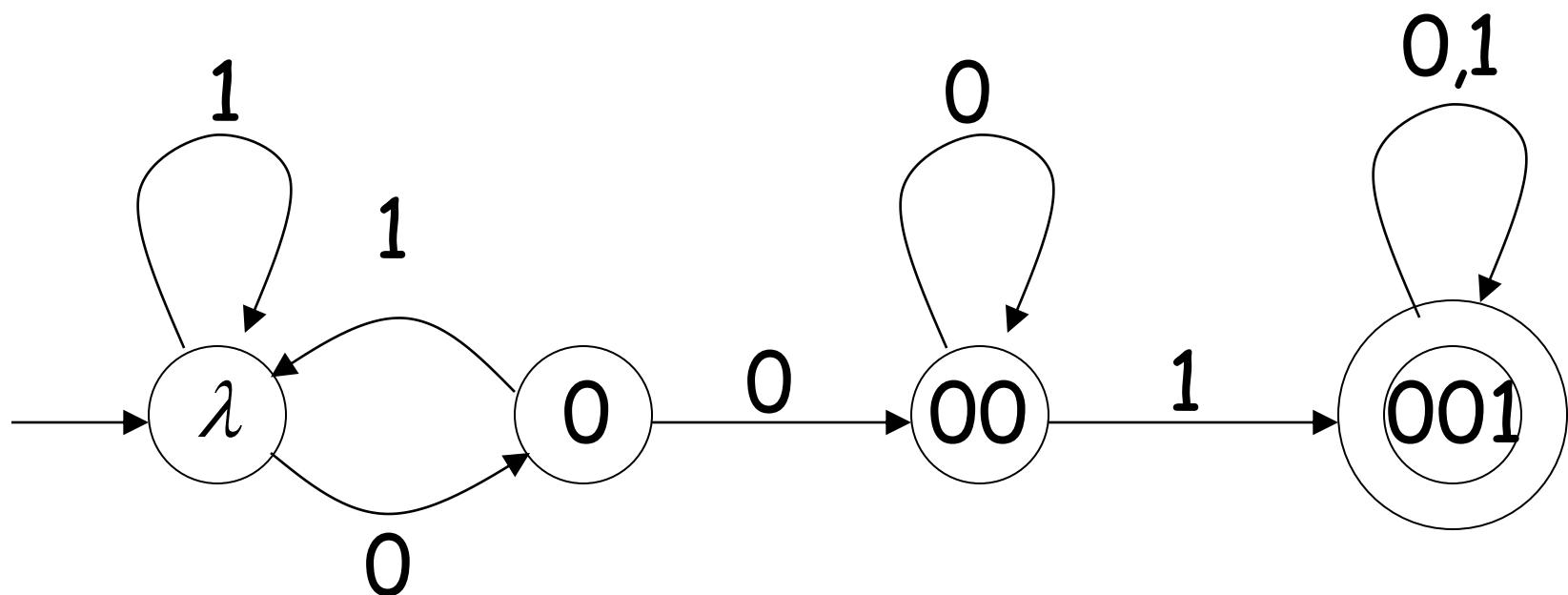
Language of the empty string

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

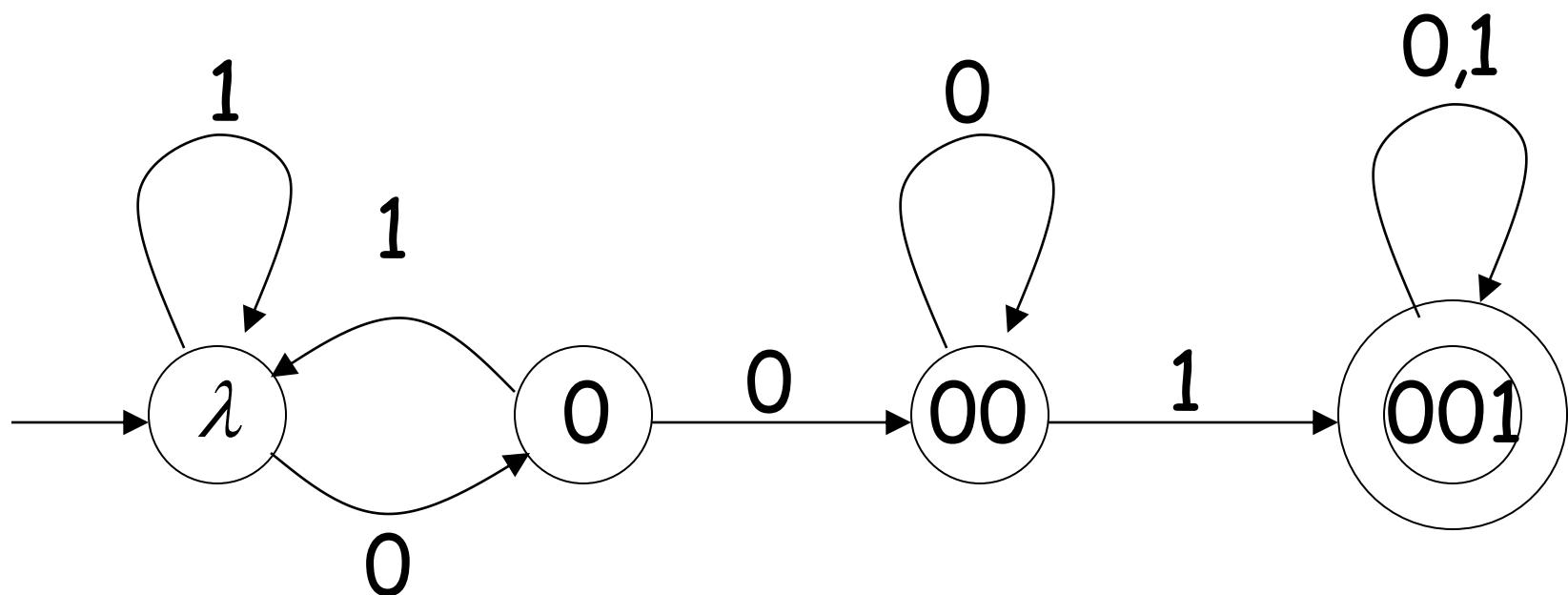
$L(M) = \{ \text{all strings with prefix } ab \}$



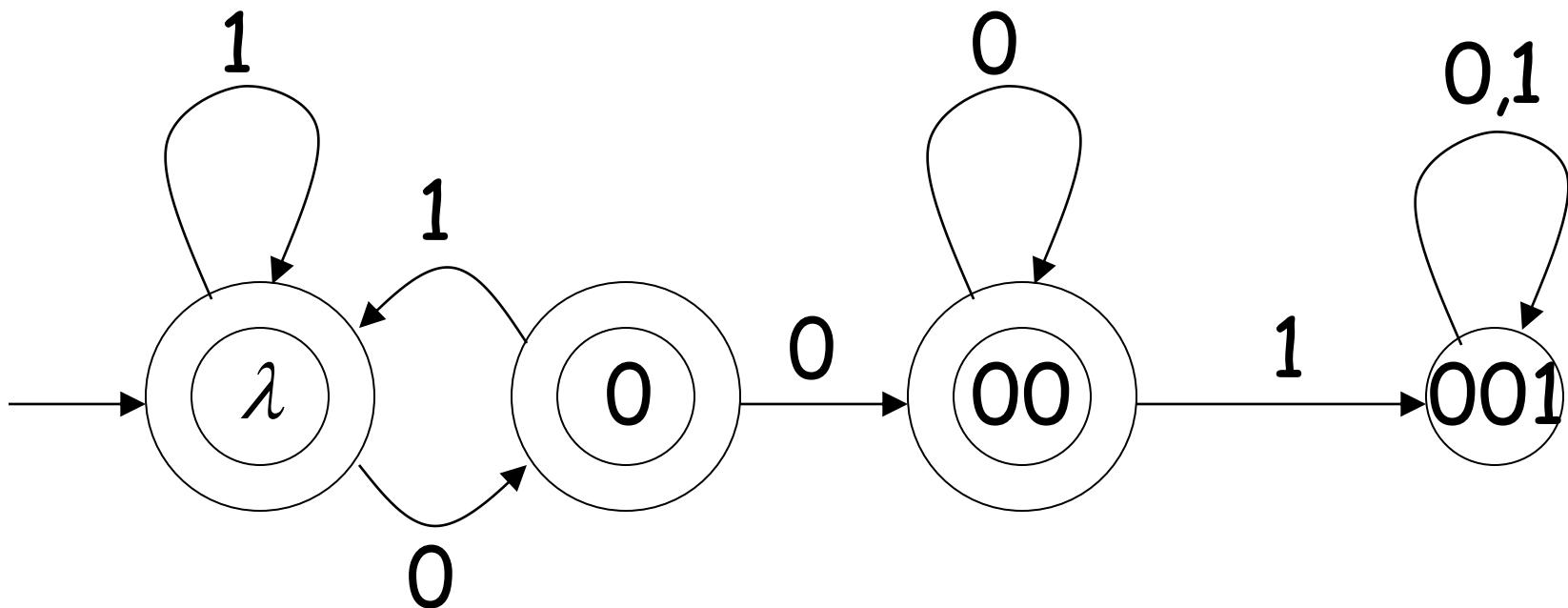
$L(M) = \{ \text{ all binary strings containing substring } 001 \}$



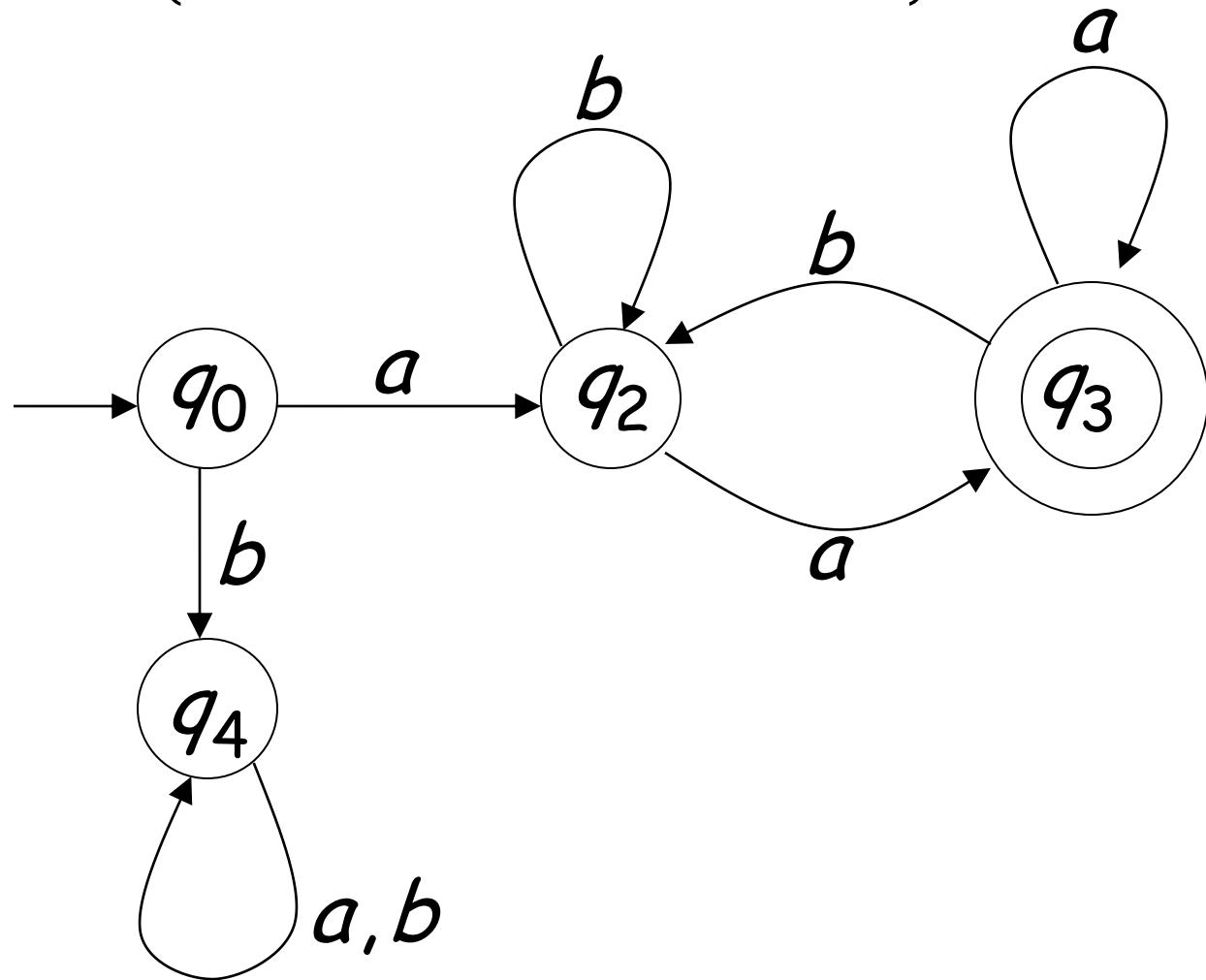
$L(M) = \{ \text{ all binary strings containing substring } 001 \}$



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



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



# Regular Languages

$L$

Definition:

A language  $M$  is **regular** if there is a DFA  $L(M) = L$  that accepts it ( )

The languages accepted by all DFAs  
form the family of **regular languages**

# Attendance 25/06/19 (Absent Nos.)

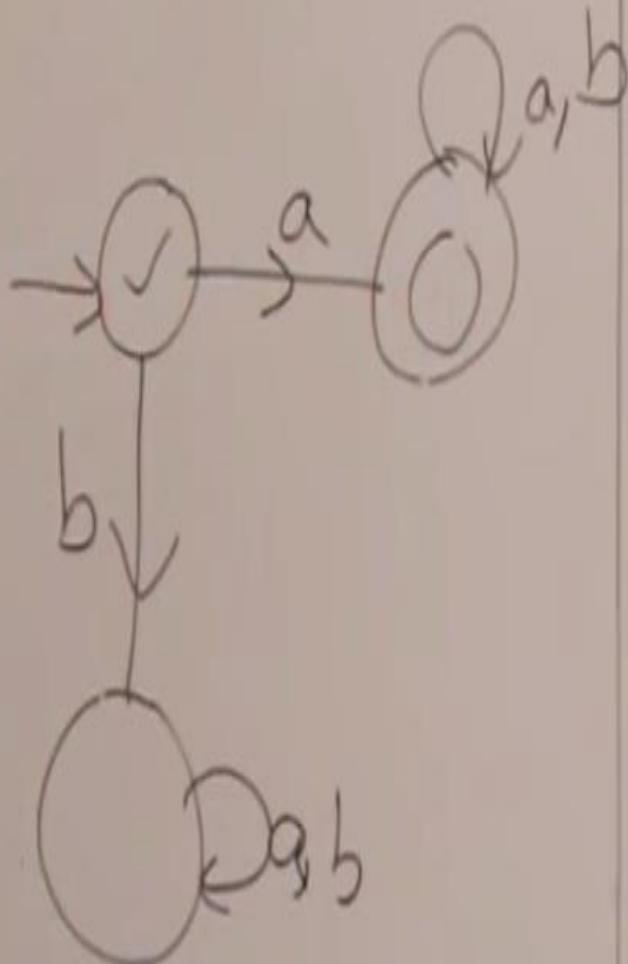
- 47,53, 62,65,71,72

# Examples

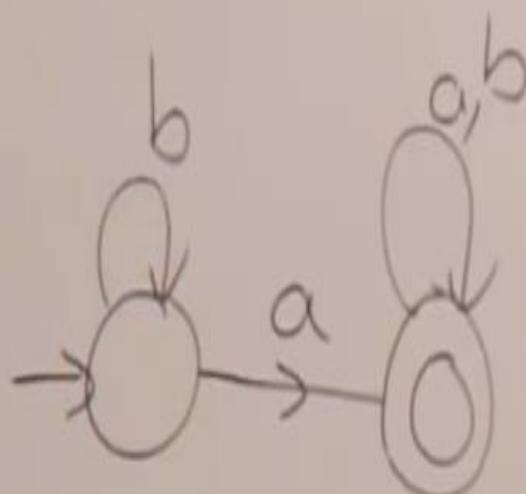
- A Finite Automaton Accepting the Language of Strings Ending in aa over the alphabet {a,b}
- An FA accepting the strings ending with *b* and *not containing aa*.
- An FA Accepting Binary Representations of Integers Divisible by 3
- An FA Accepting Strings That Contain Either ab or bba.

- FA which read string made of {0,1} and accepts those string which end with 00 or 11.(Insem-2017, 4 Marks)
- Design FA Accepting Language of Strings Ending in b and not containing the substring aa.
- An FA accepting Binary Representations of Integers Divisible by 3.

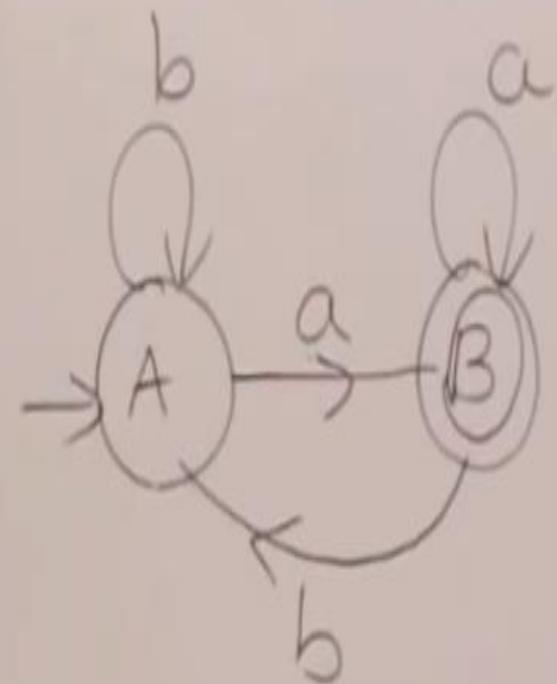
Starts with a



Containing a



Ends with a



bbbab  a

# Regular Languages

Definition:

A language  $L$  is regular if there is a DFA  $M$  that accepts it ( $L(M) = L$ )

The languages accepted by all DFAs form the family of regular languages

**A language  $L$  is **regular** if it is  
**recognized** by a deterministic  
finite automaton (DFA),  
i.e. if there is a DFA  $M$  such  
that  $L = L(M)$ .**

$L = \{ w \mid w \text{ contains } 001\}$  is regular

$L = \{ w \mid w \text{ has an even number of } 1's\}$  is regular

## Example regular languages:

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

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

{ all strings in  $\{a,b\}^*$  with prefix  $ab$  }

{ all binary strings without substring 001}

{ $x : x \in \{1\}^*$  and  $x$  is even}

{ }     $\{\lambda\}$      $\{a,b\}^*$

There exist automata that accept these languages (see previous slides).

There exist languages which are not Regular:

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

$$\text{ADDITION} = \{x + y = z : x = 1^n, y = 1^m, z = 1^k, n + m = k\}$$

There is no DFA that accepts these languages  
(we will prove this in a later class)

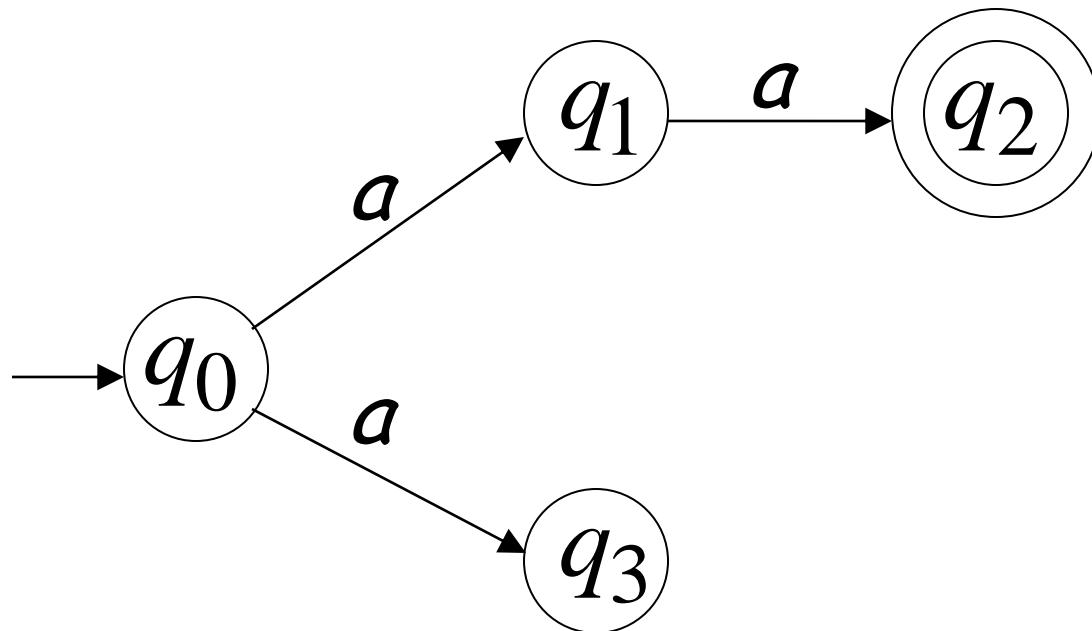
# Recall

- Examples of DFA
- NFA

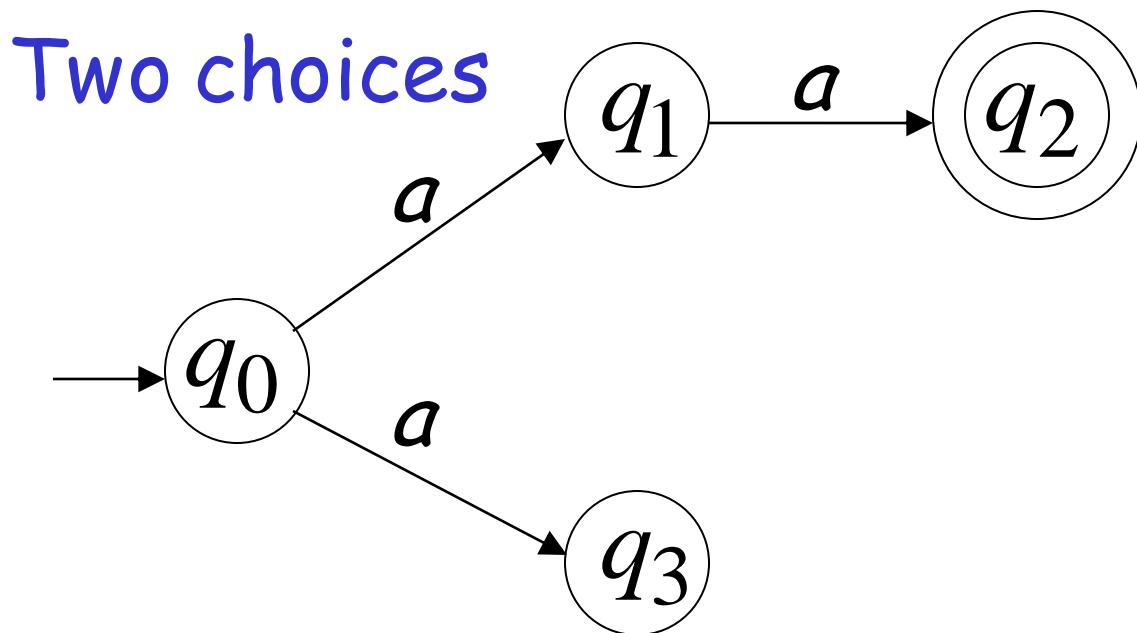
# Non-Deterministic Finite Automata

# Nondeterministic Finite Automaton (NFA)

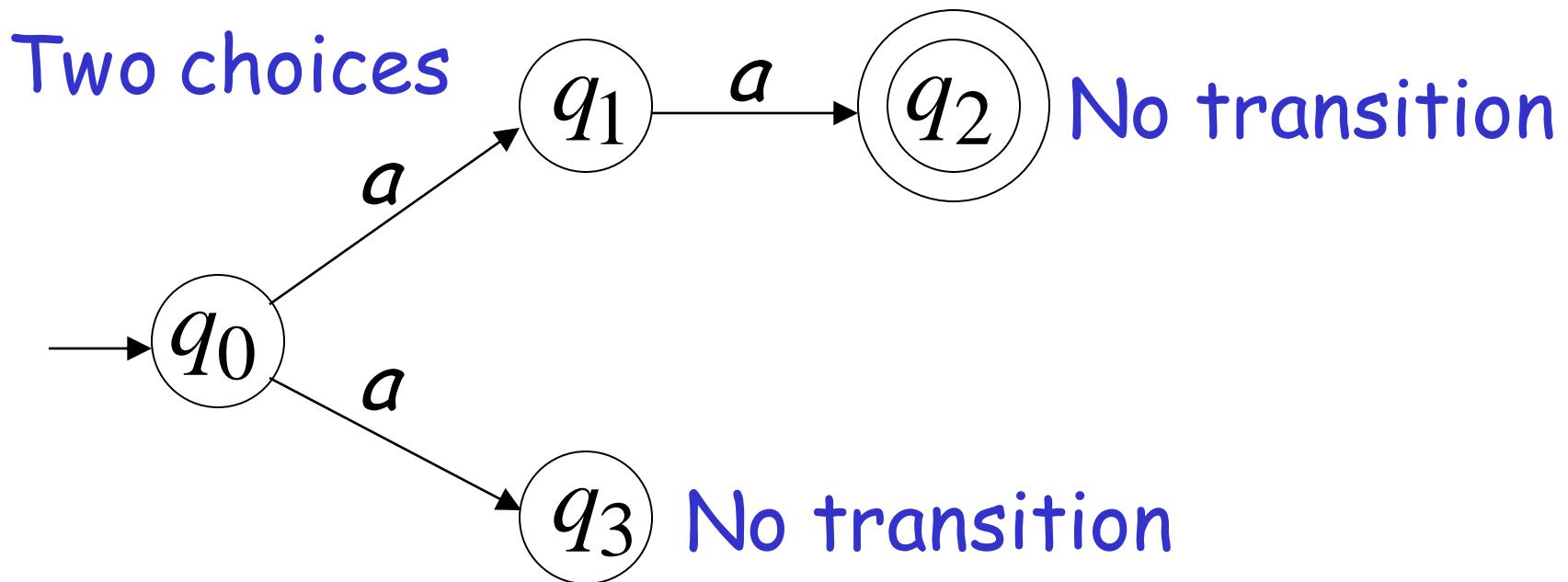
Alphabet =  $\{a\}$



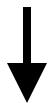
Alphabet =  $\{a\}$



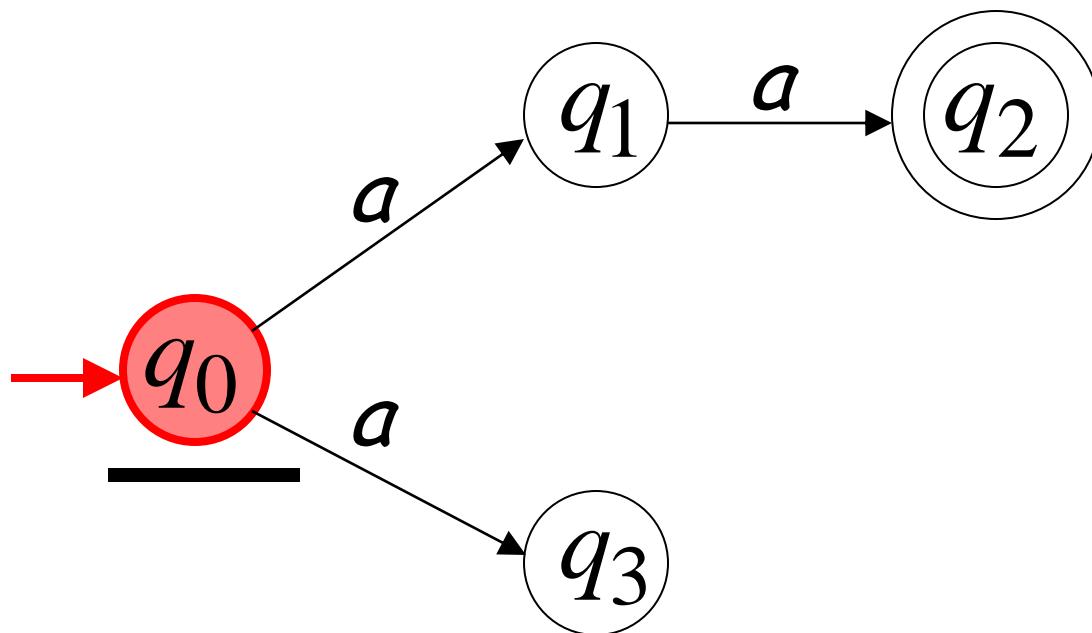
Alphabet =  $\{a\}$



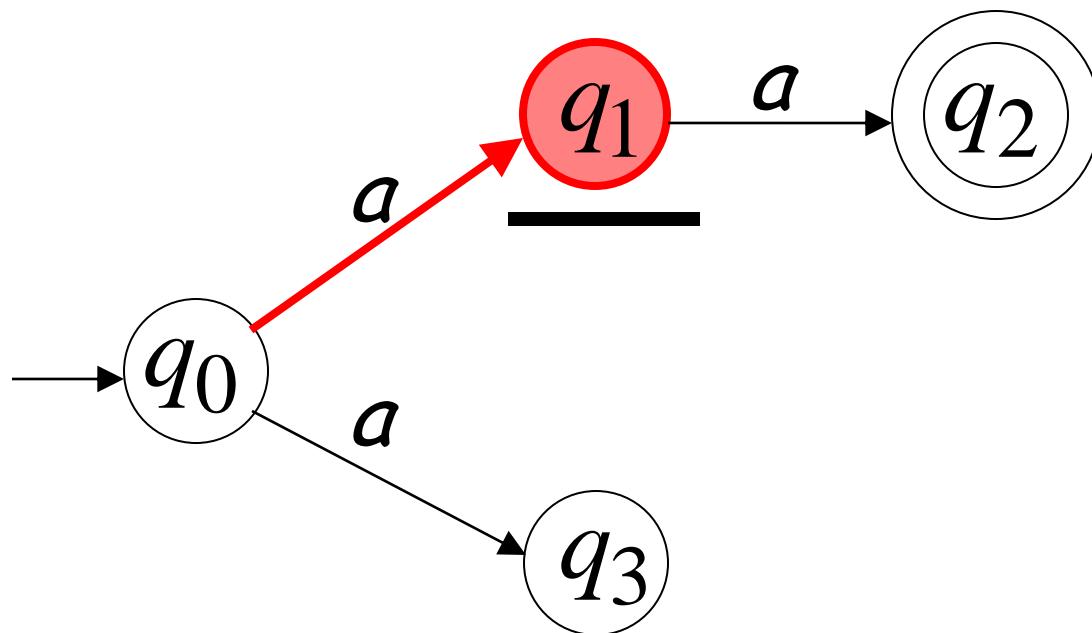
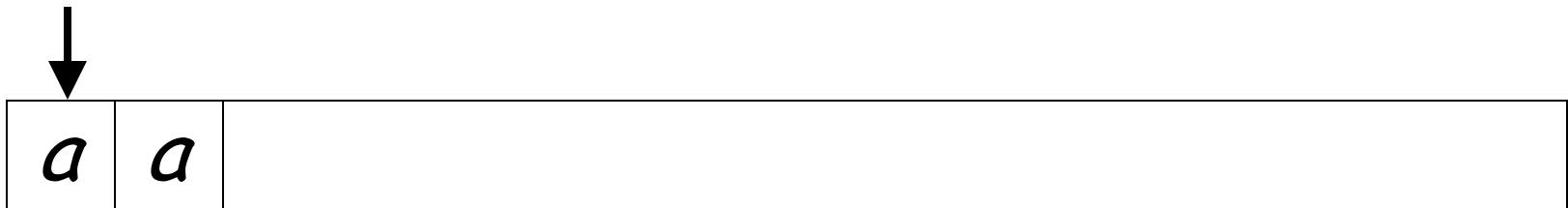
# First Choice



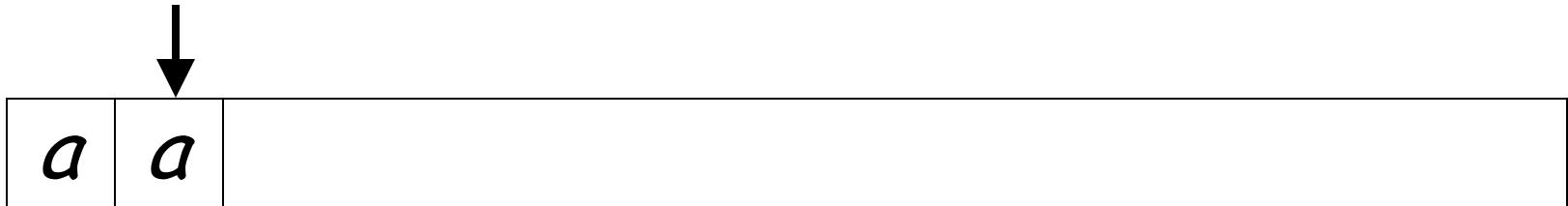
$a$	$a$	
-----	-----	--



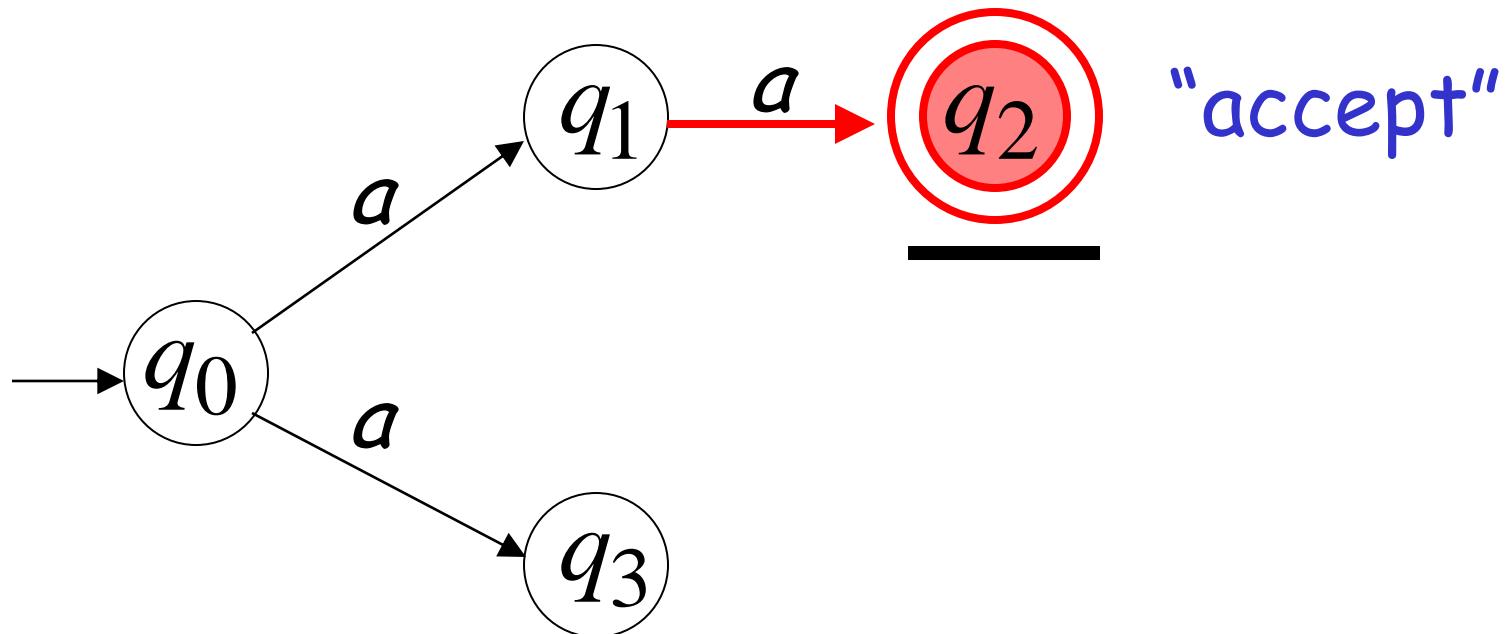
# First Choice



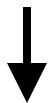
# First Choice



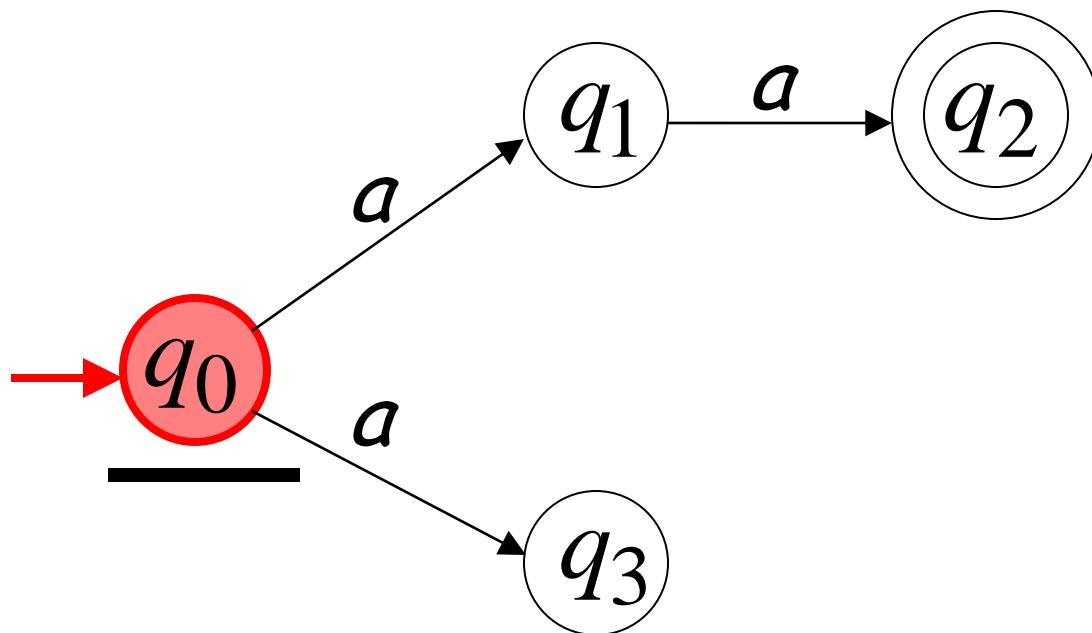
All input is consumed



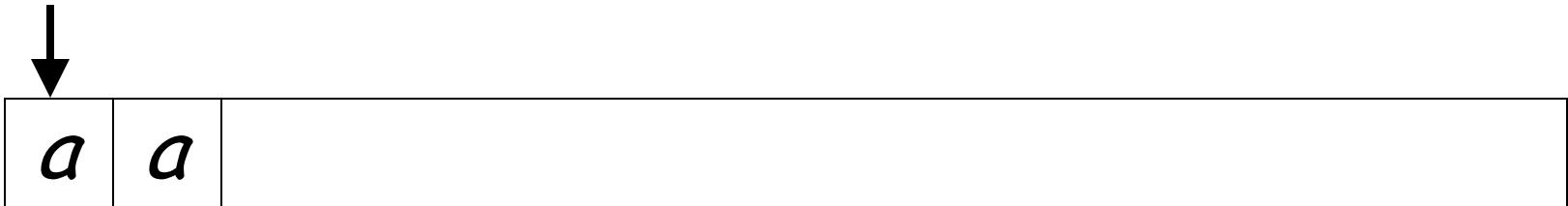
# Second Choice



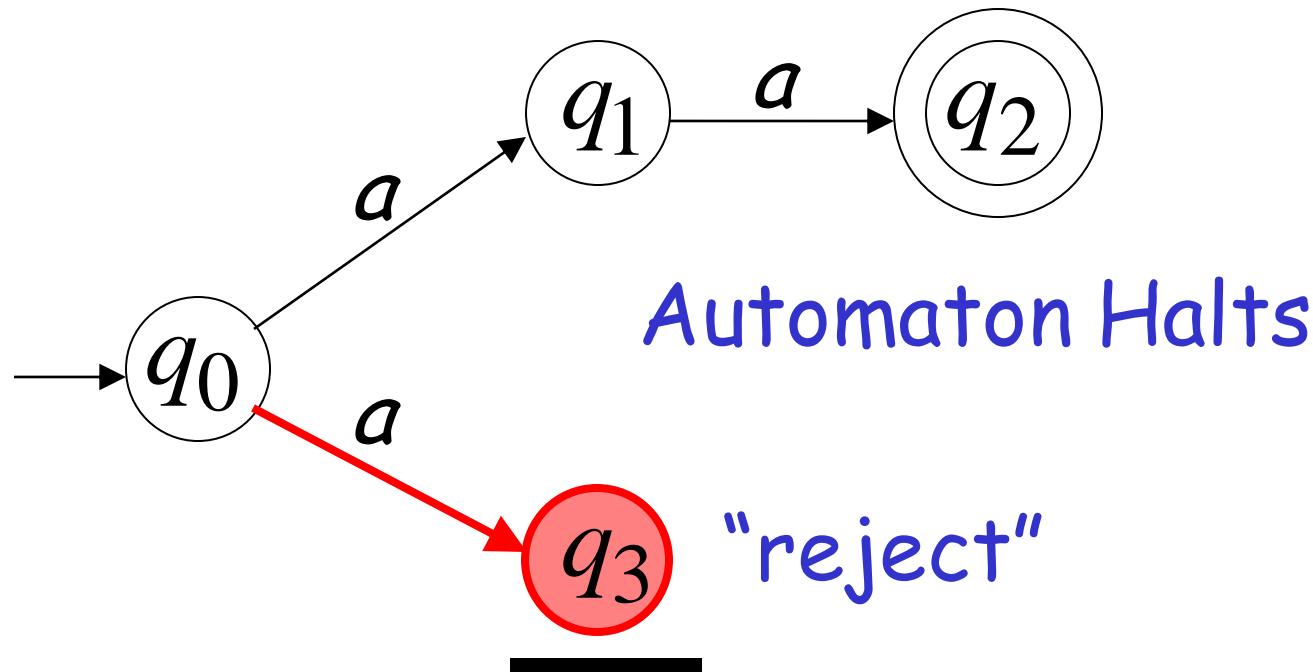
$a$	$a$	
-----	-----	--



# Second Choice



Input cannot be consumed

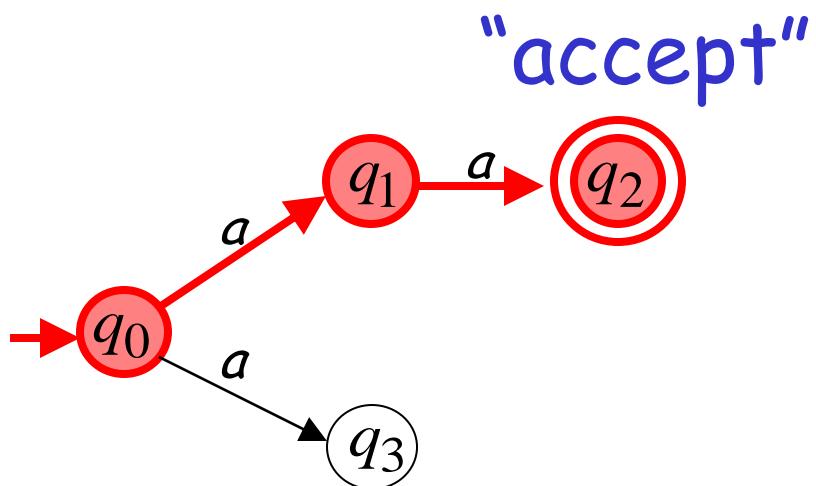


An NFA accepts a string:

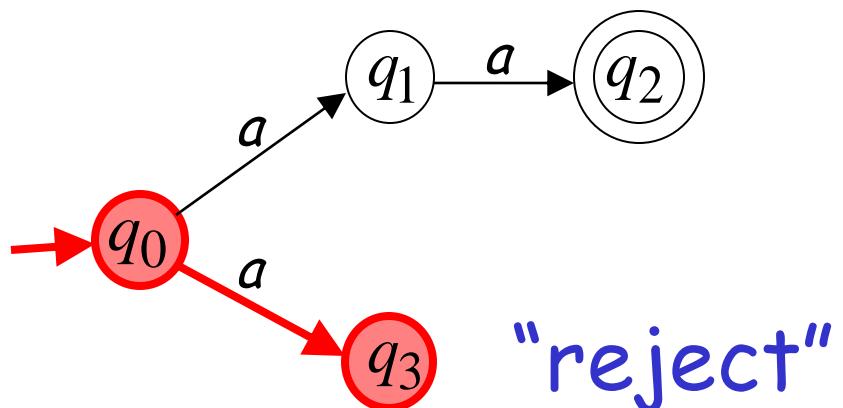
if there is a computation of the NFA  
that accepts the string

i.e., all the input string is processed and the  
automaton is in an accepting state

$aa$  is accepted by the NFA:



because this computation accepts  $aa$

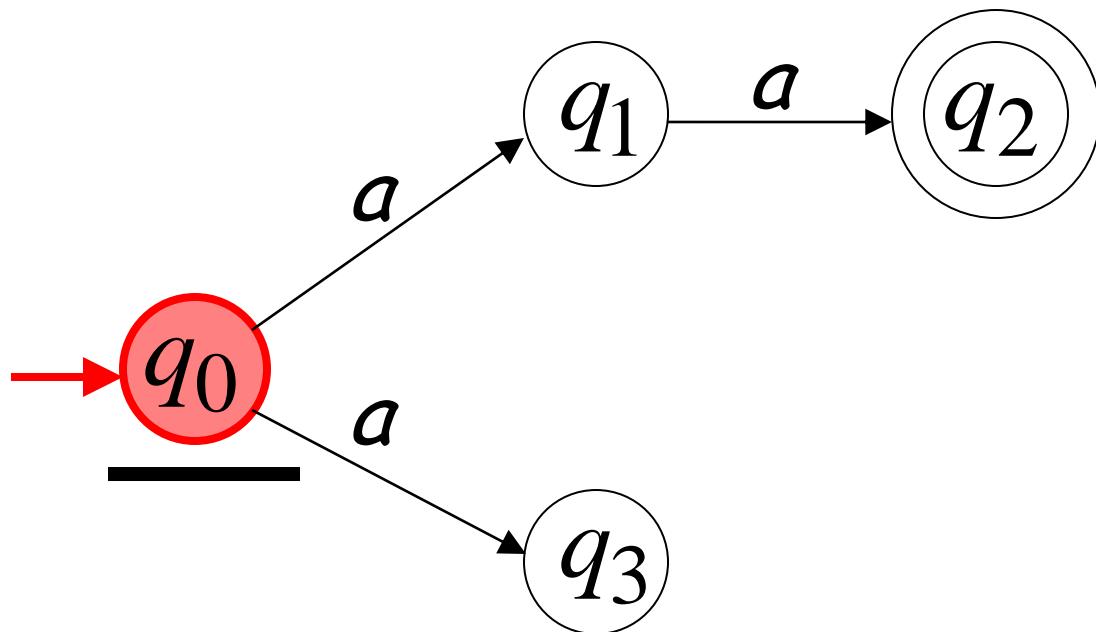
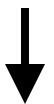


this computation is ignored

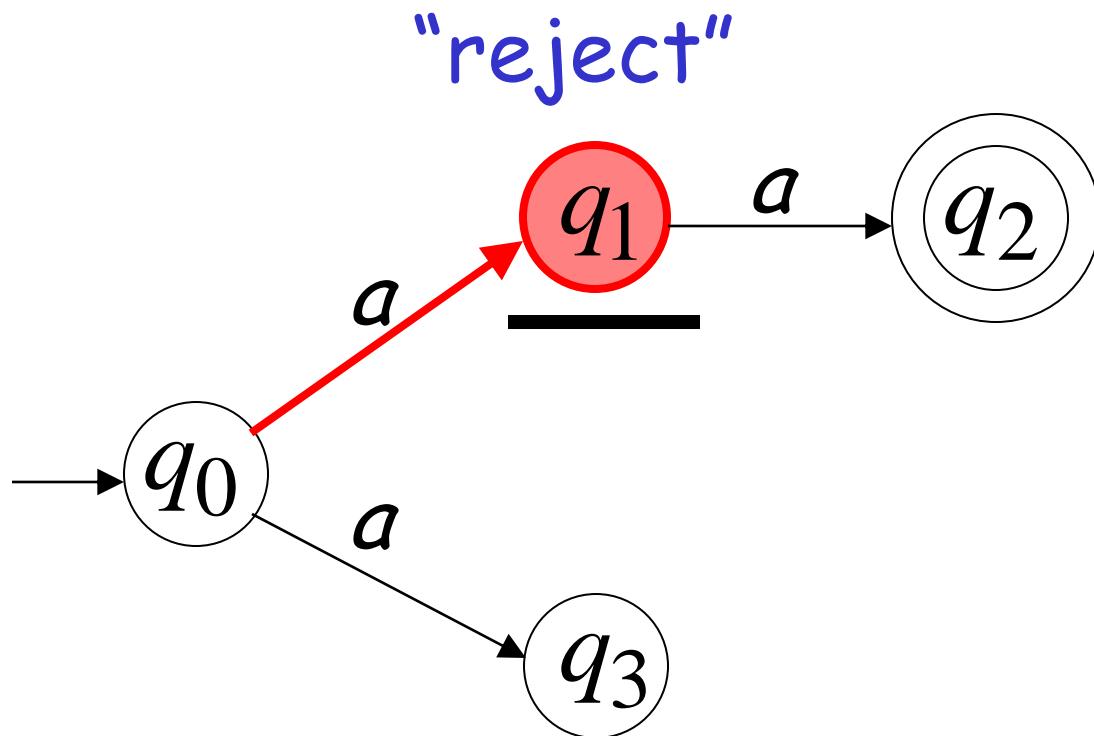
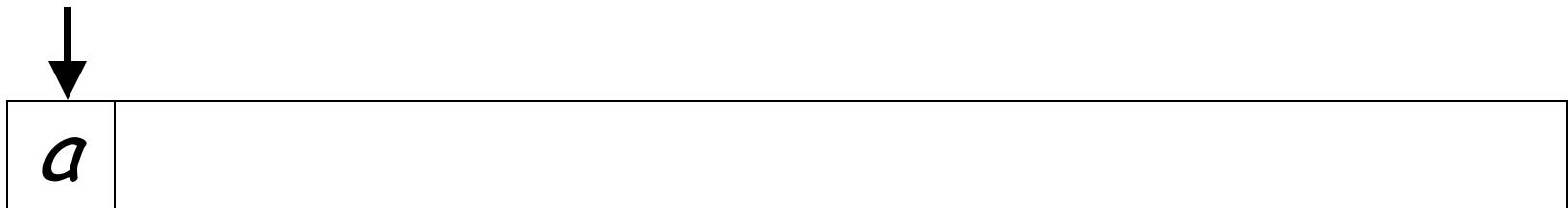
Absent No 27/6/19

2,11,16,25,27,28,34,37,38,42,43,53,54,  
59,65,69,71,72,74

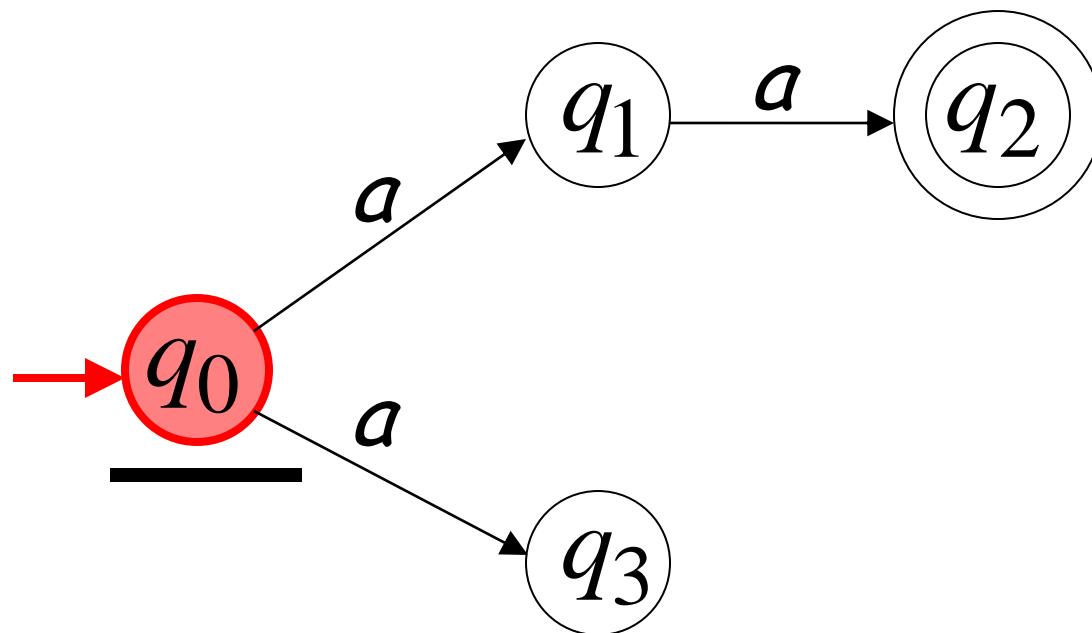
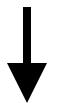
# Rejection example



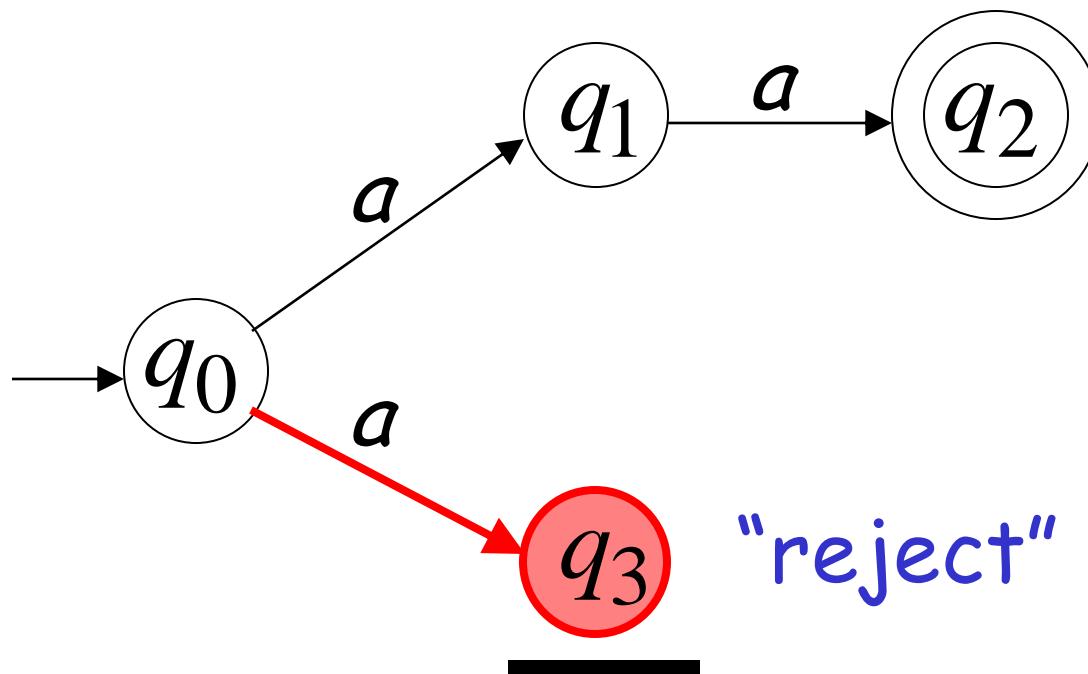
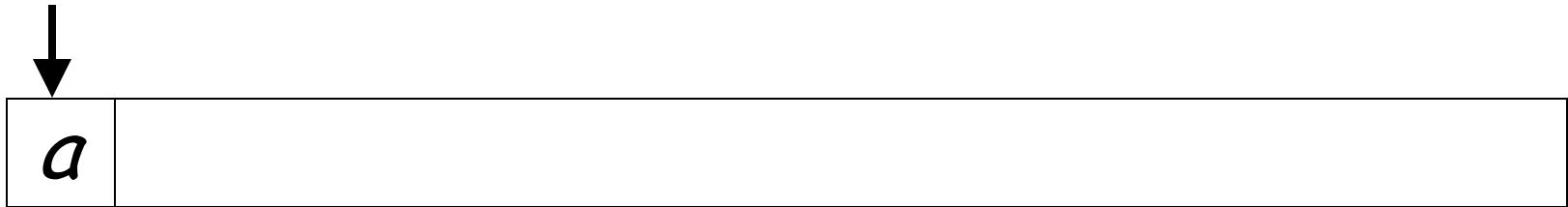
# First Choice



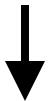
# Second Choice



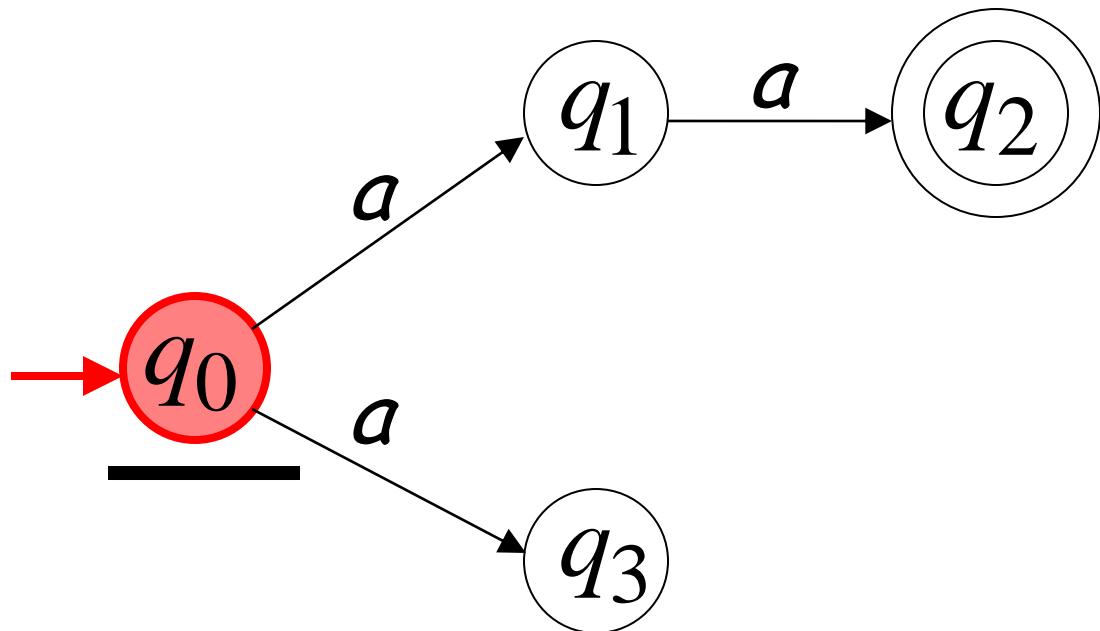
# Second Choice



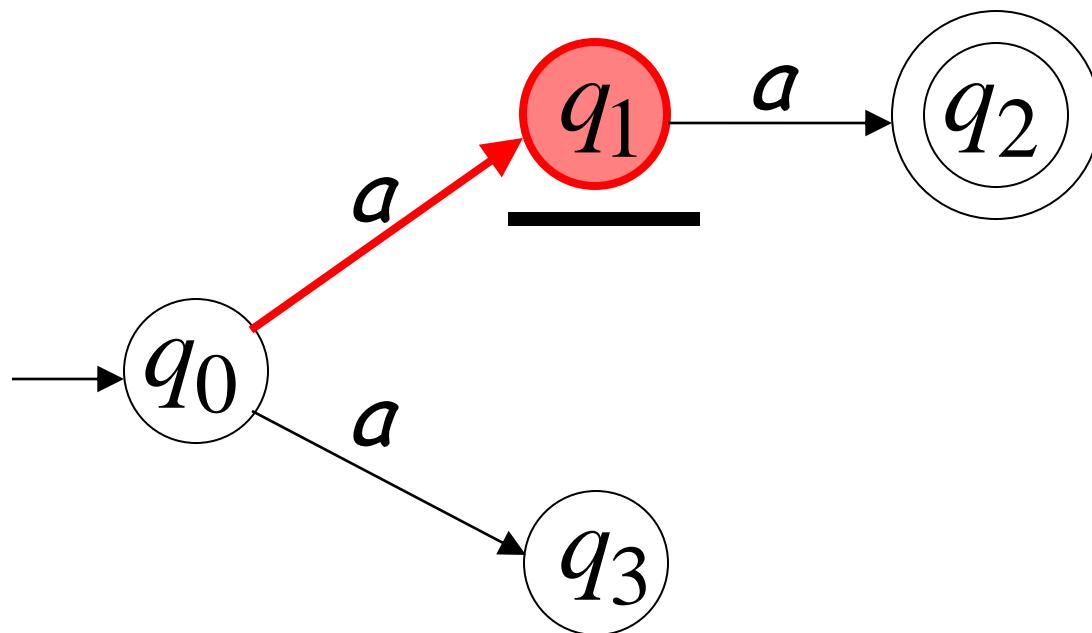
# Another Rejection example



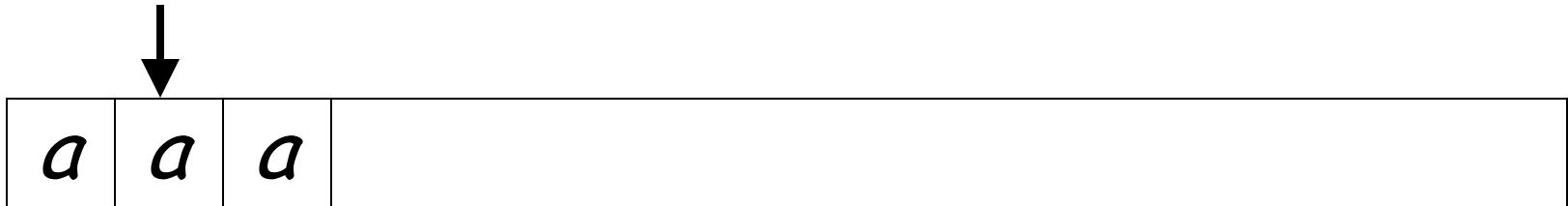
$a$	$a$	$a$	
-----	-----	-----	--



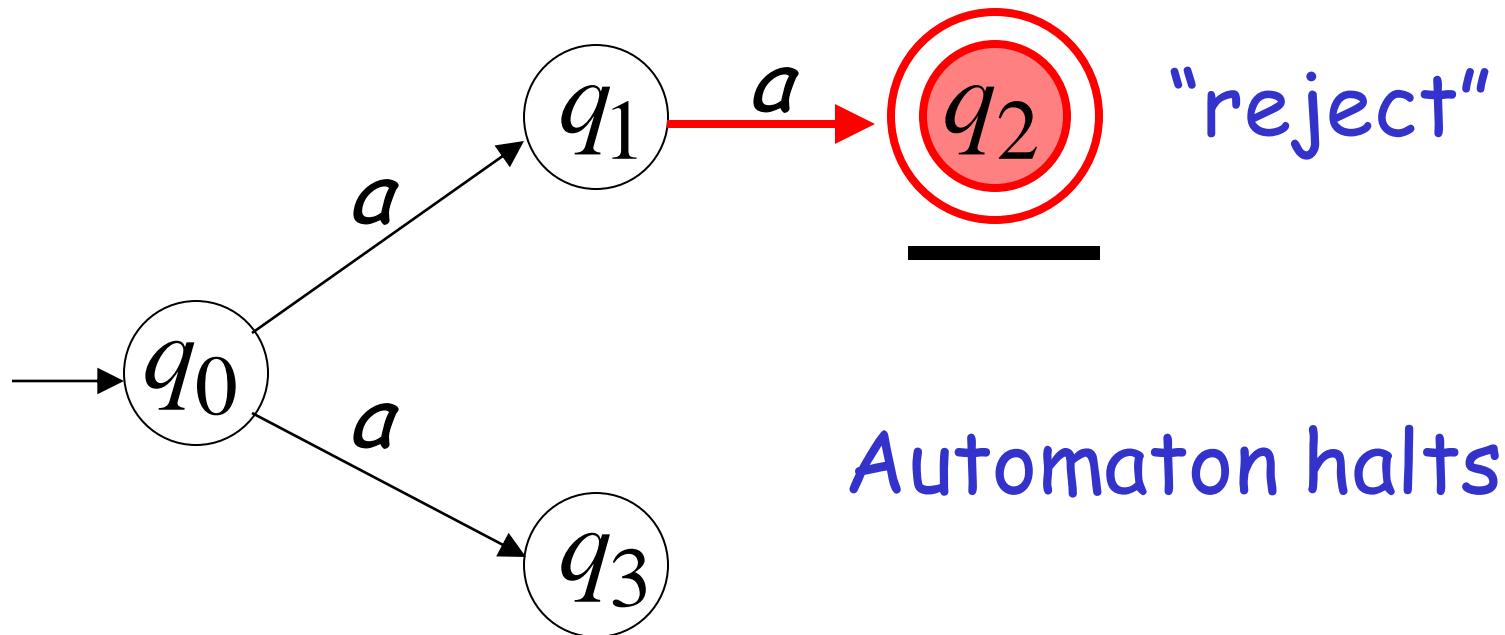
# First Choice



# First Choice



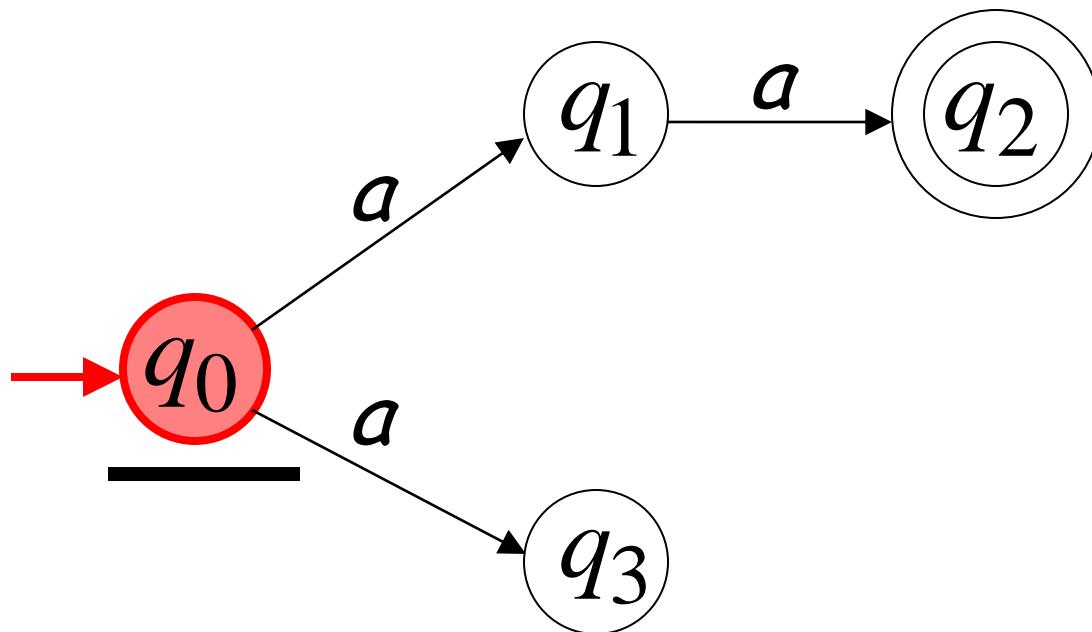
Input cannot be consumed



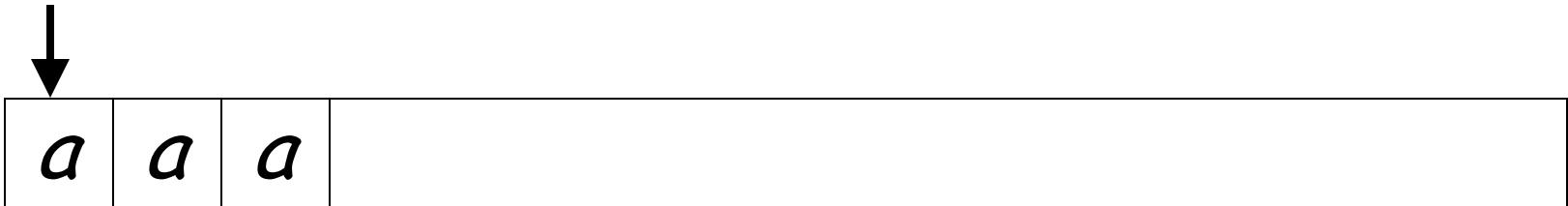
# Second Choice



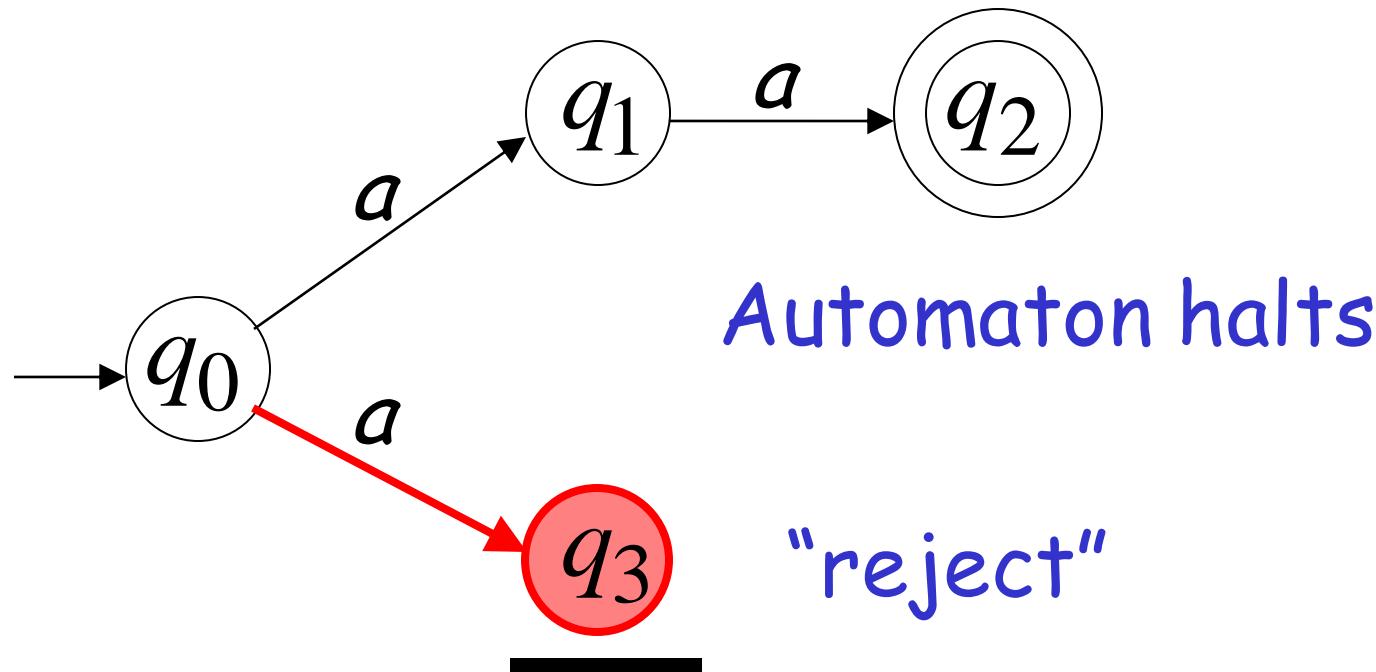
$a$	$a$	$a$	
-----	-----	-----	--



# Second Choice



Input cannot be consumed



An NFA rejects a string:

if there is no computation of the NFA  
that accepts the string.

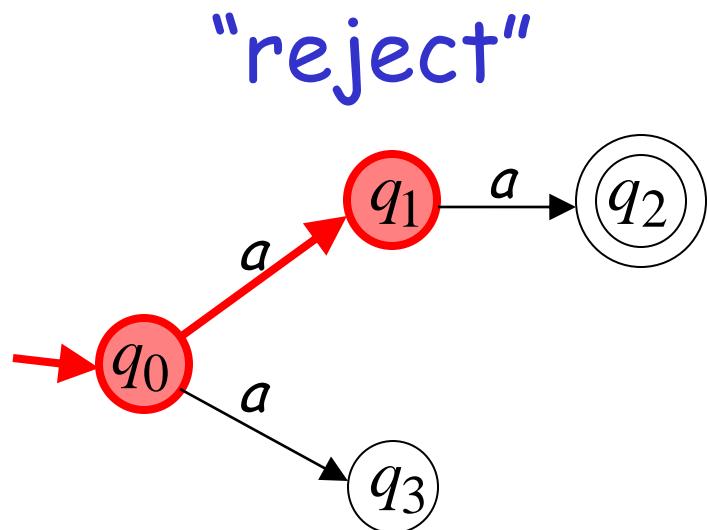
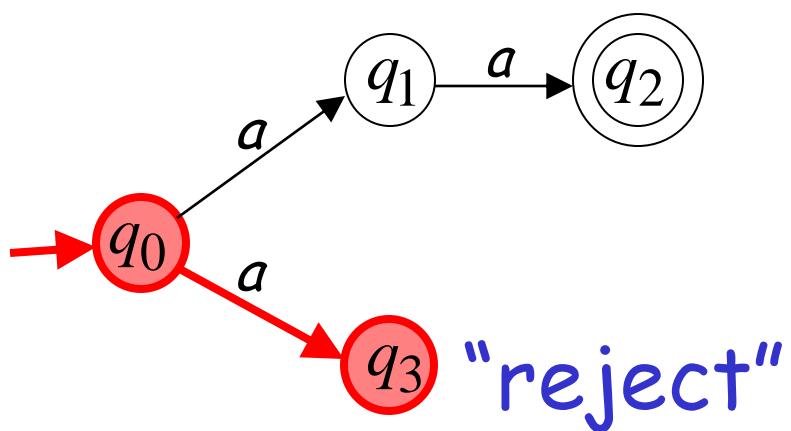
For each computation:

- All the input is consumed and the automaton is in a non final state

OR

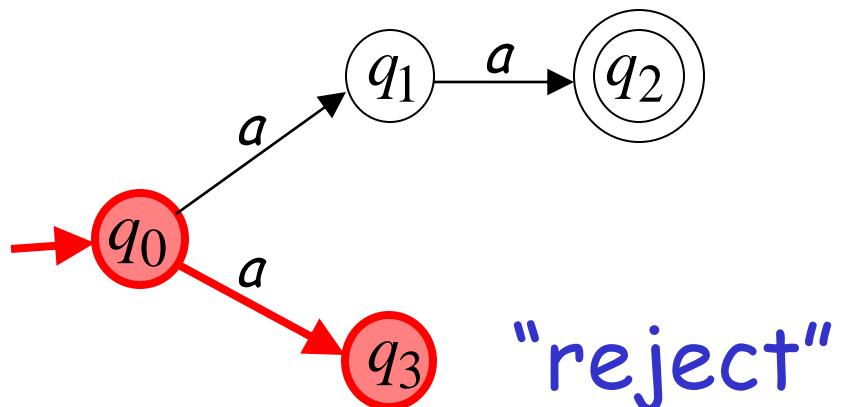
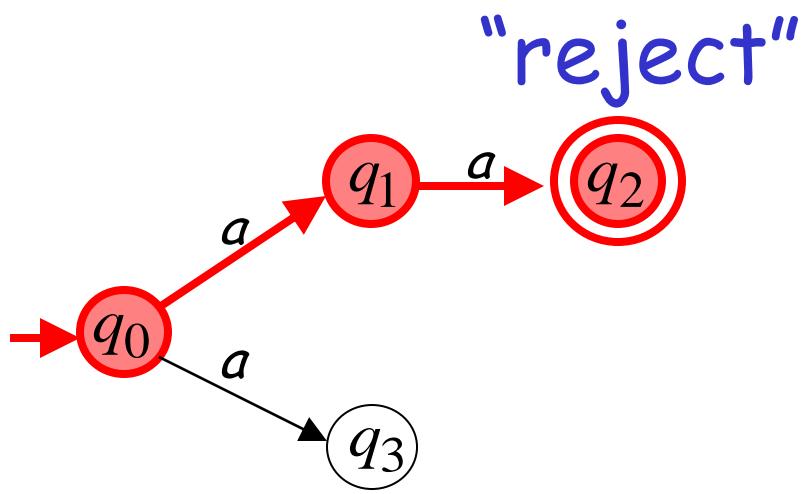
- The input cannot be consumed

a is rejected by the NFA:



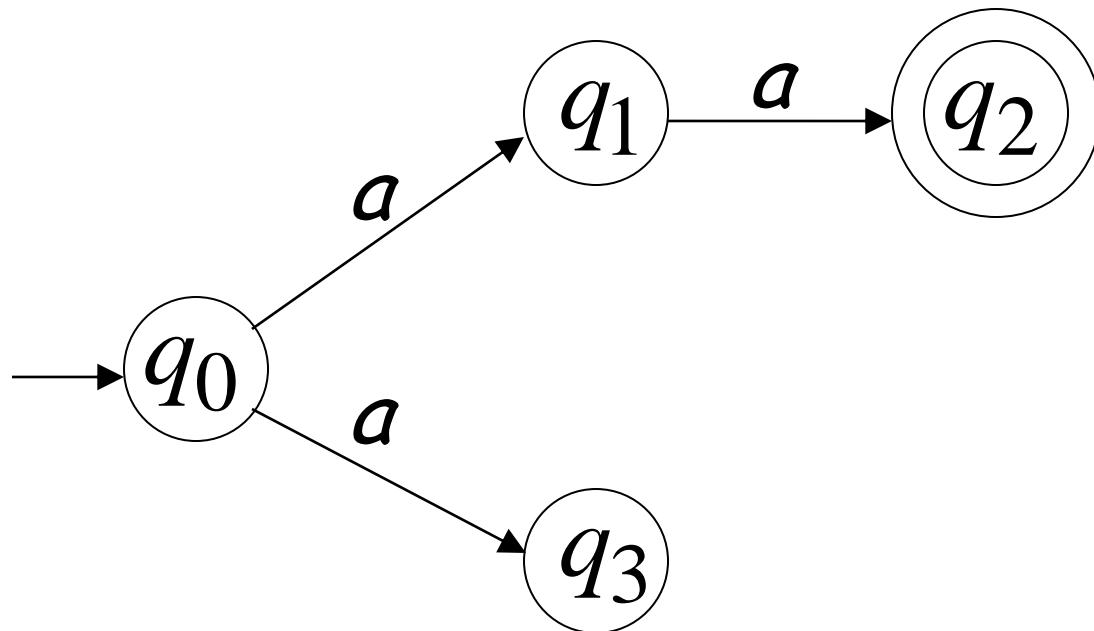
All possible computations lead to rejection

**aaa** is rejected by the NFA:

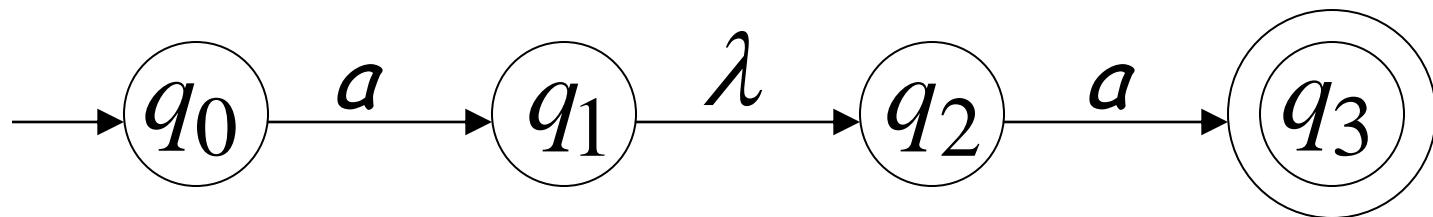


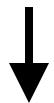
All possible computations lead to rejection

Language accepted:  $L = \{aa\}$

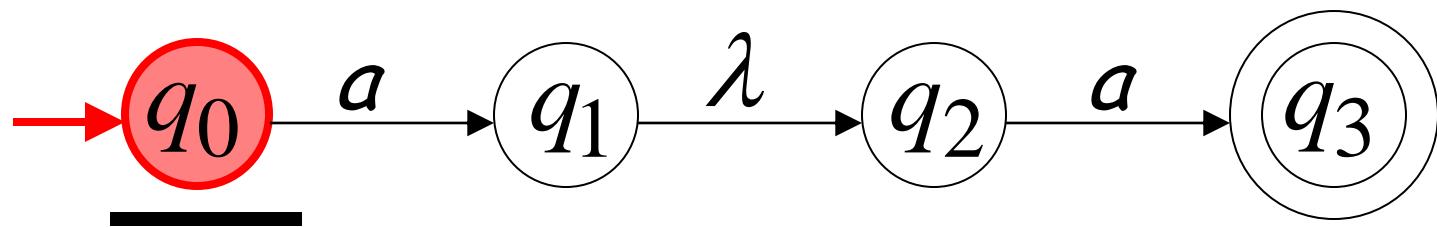


# Lambda Transitions



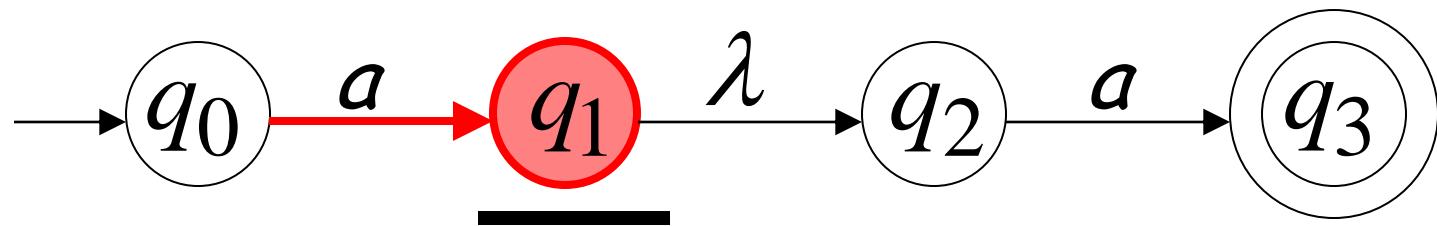


$a$	$a$	
-----	-----	--

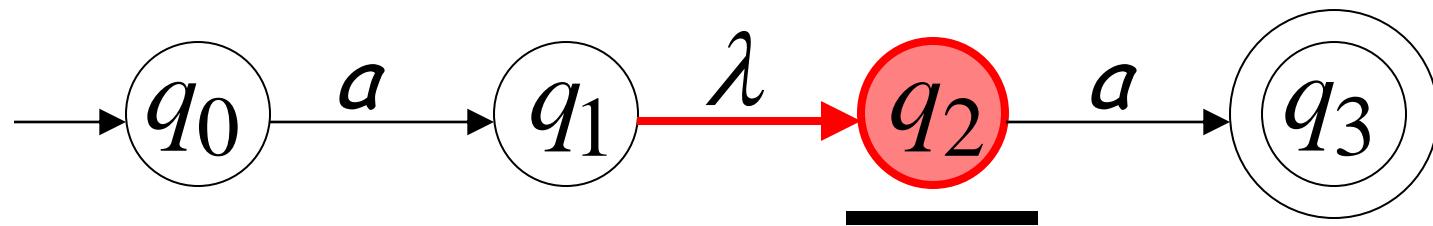
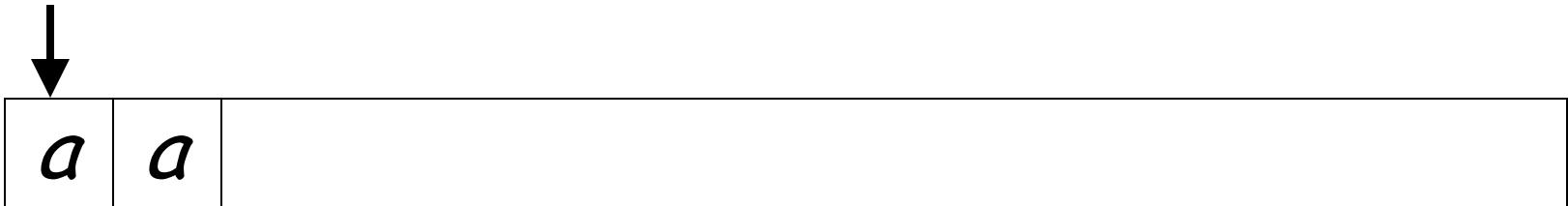


↓

$a$	$a$	
-----	-----	--



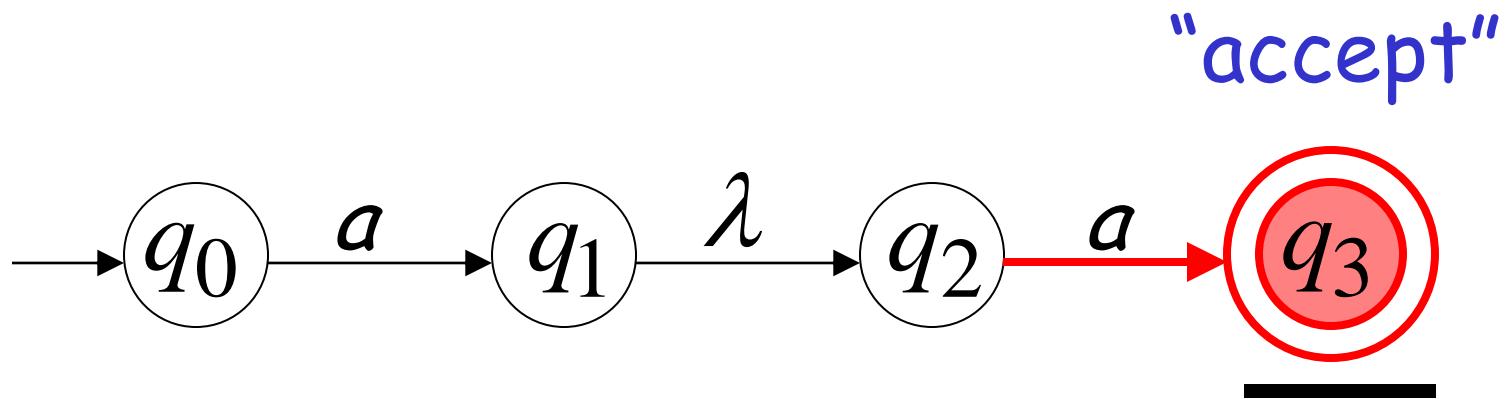
input tape head does not move



all input is consumed

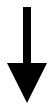


$a$	$a$	
-----	-----	--

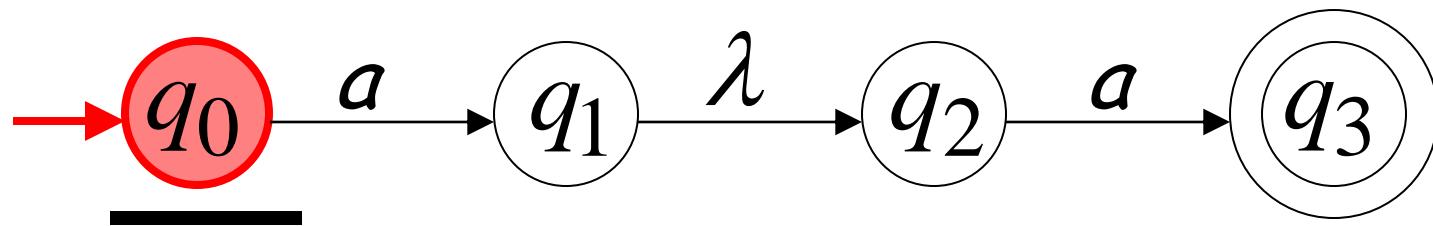


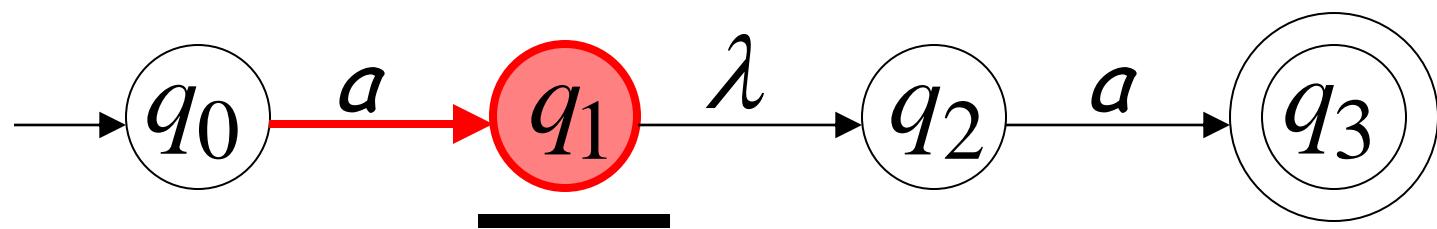
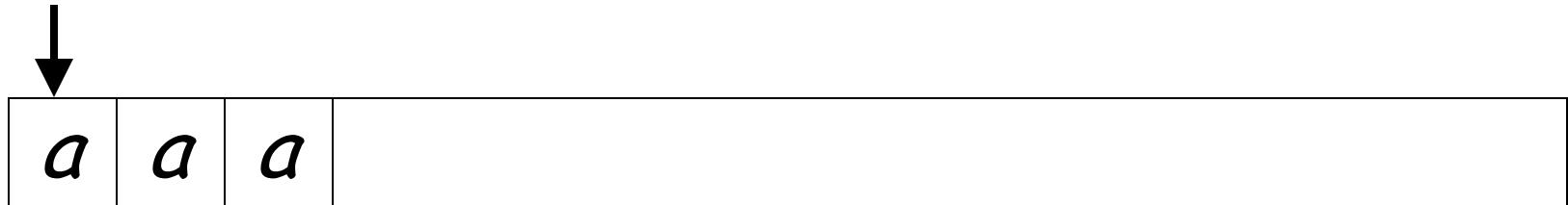
String  $aa$  is accepted

# Rejection Example

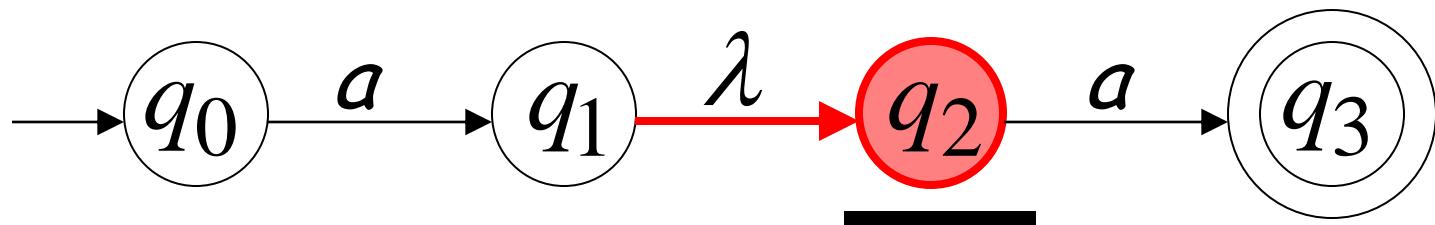
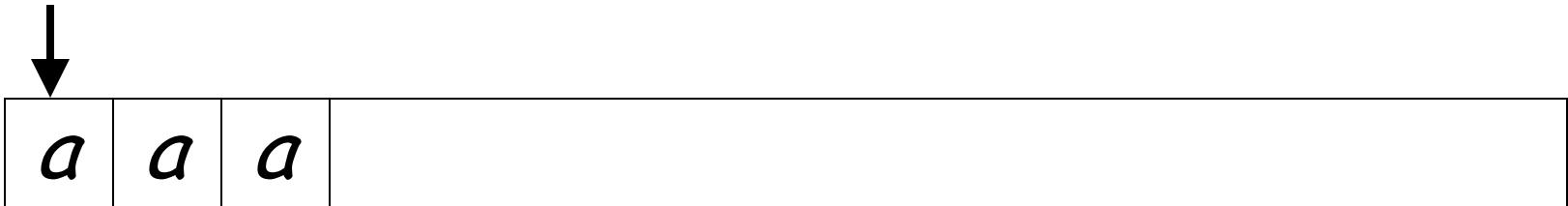


a	a	a	
---	---	---	--





(read head doesn't move)



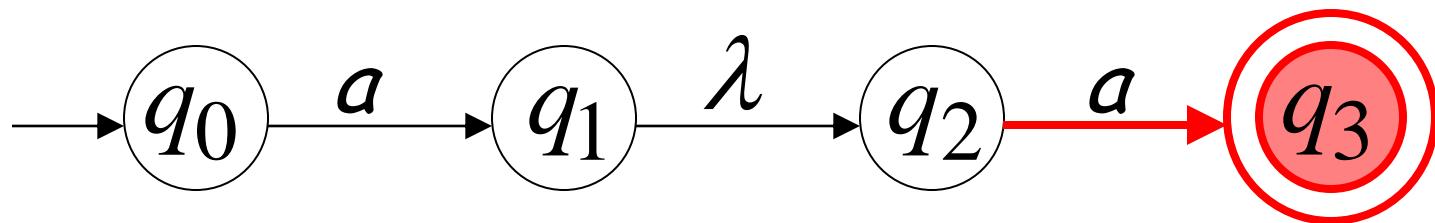
# Input cannot be consumed



a	a	a	
---	---	---	--

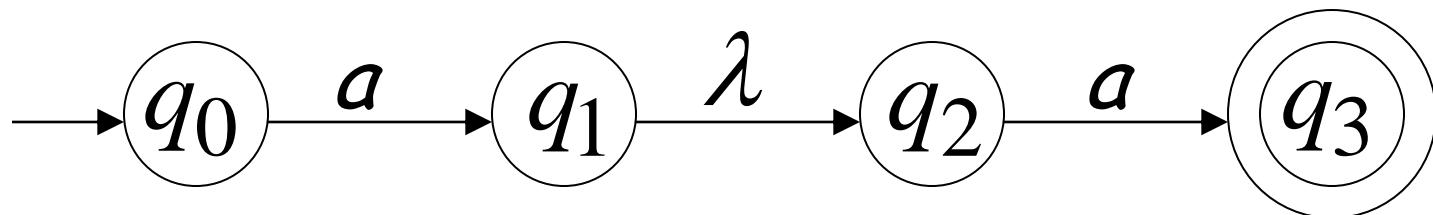
Automaton halts

"reject"

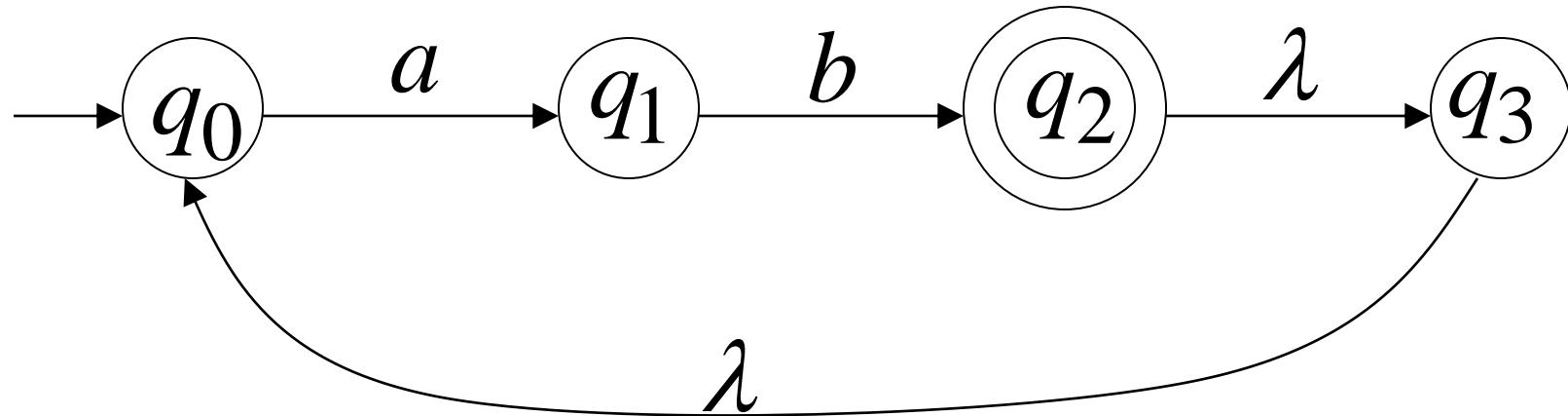


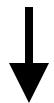
String aaa is rejected

Language accepted:  $L = \{aa\}$

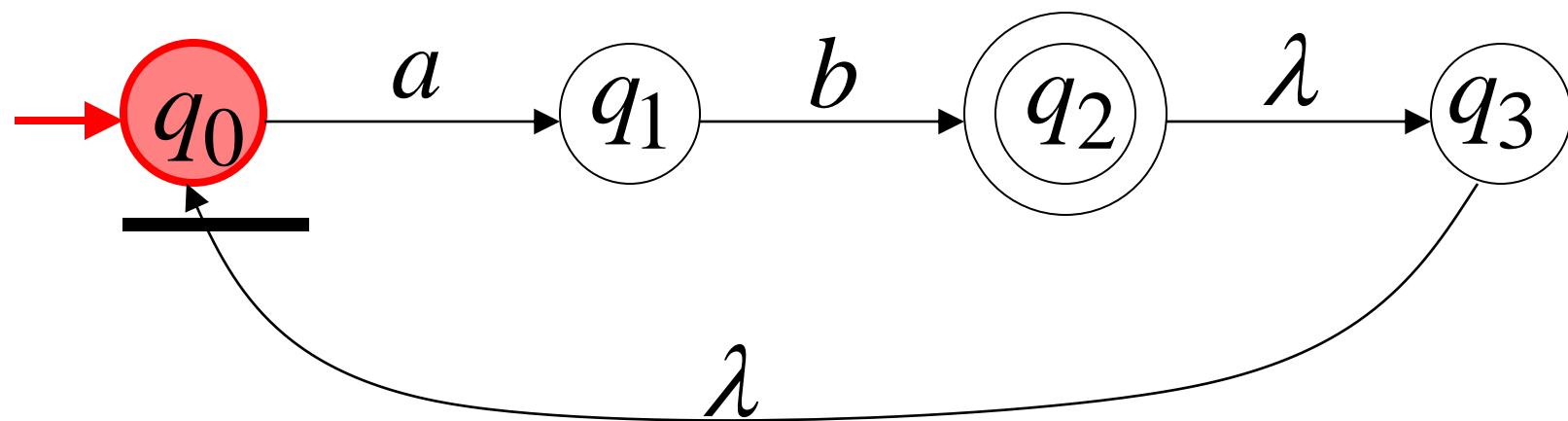


# Another NFA Example

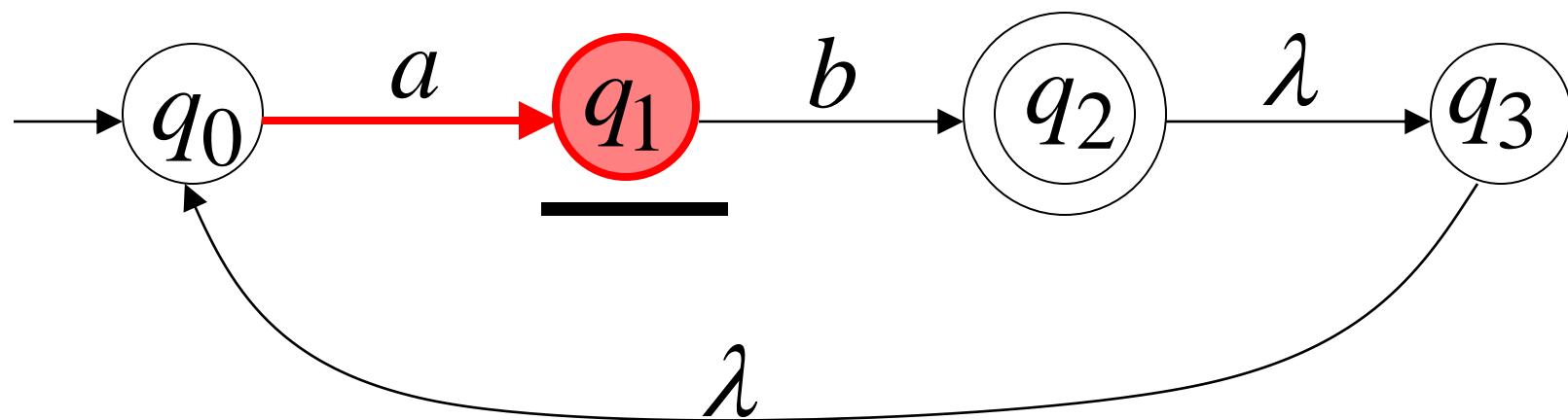




$a$	$b$	
-----	-----	--

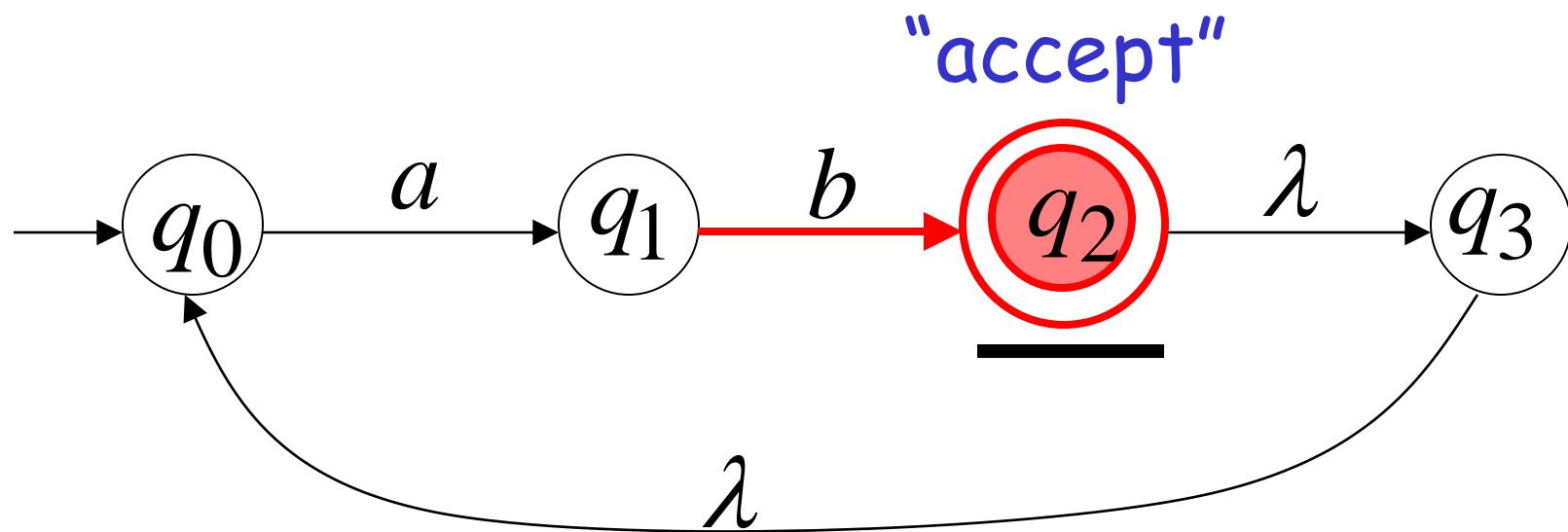


$a$	$b$	
-----	-----	--



↓

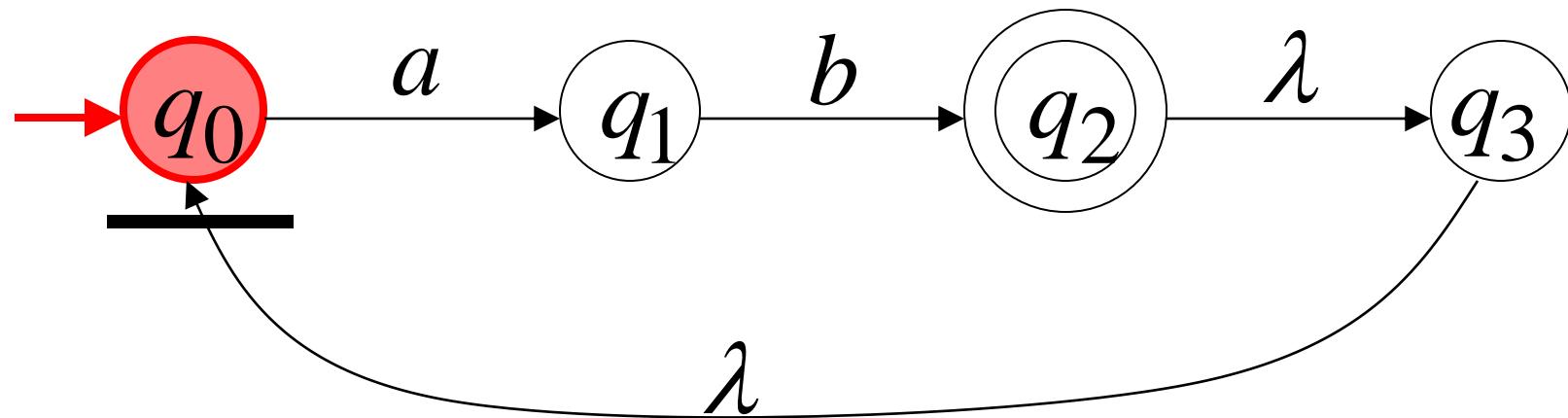
$a$	$b$	
-----	-----	--



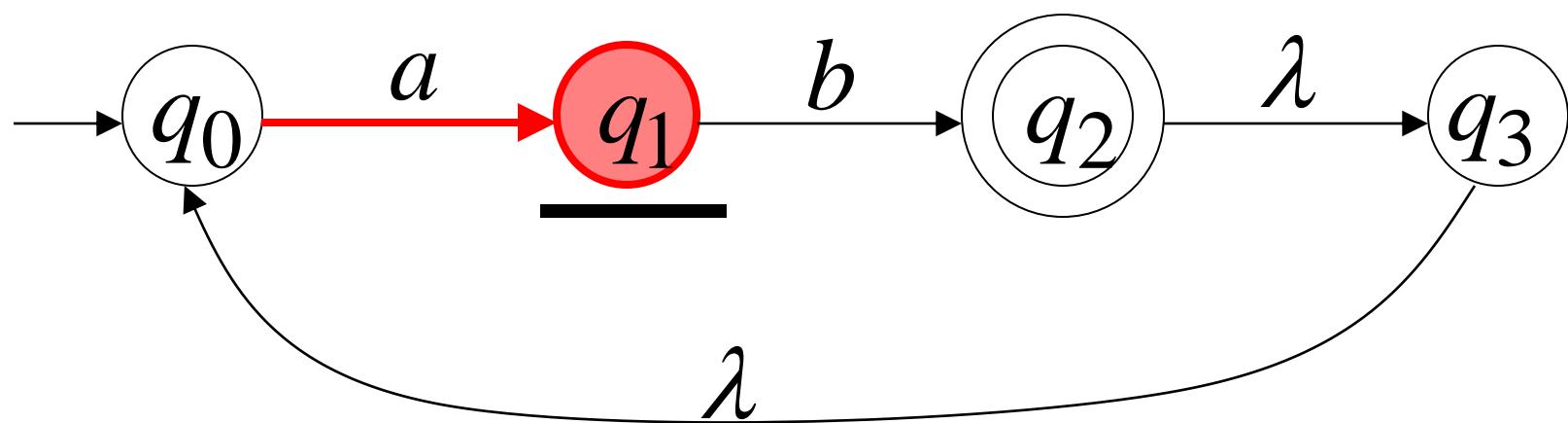
## Another String



$a$	$b$	$a$	$b$	
-----	-----	-----	-----	--

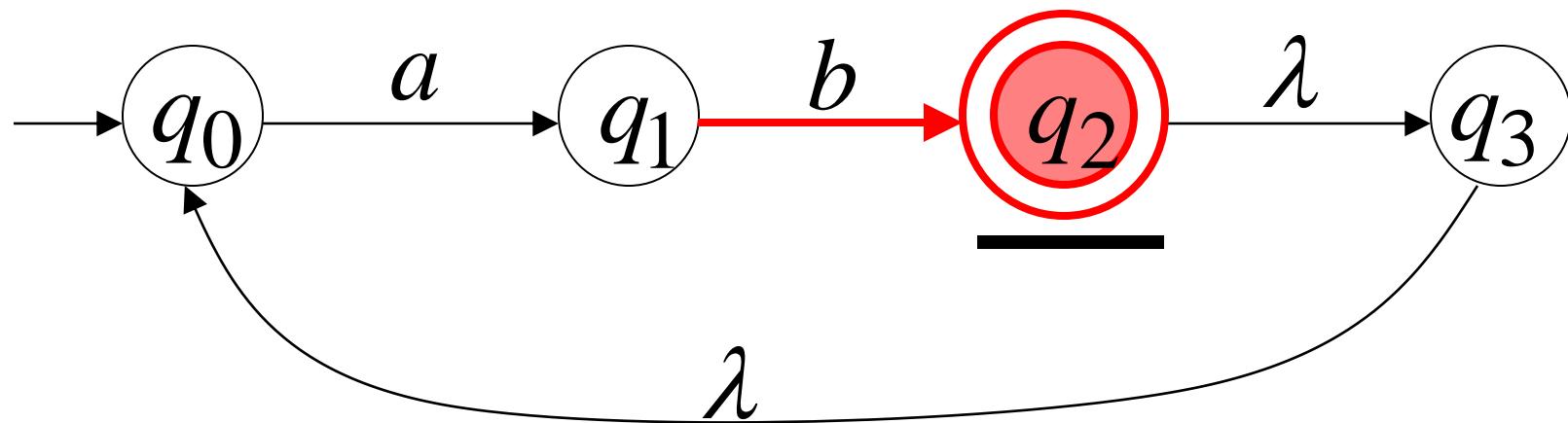


$a$	$b$	$a$	$b$	

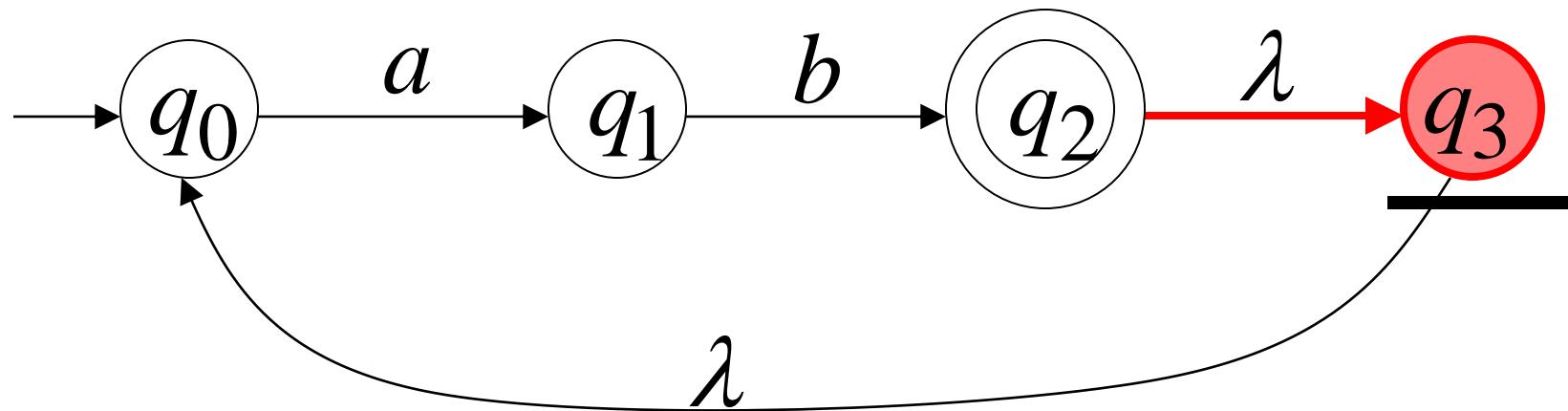


↓

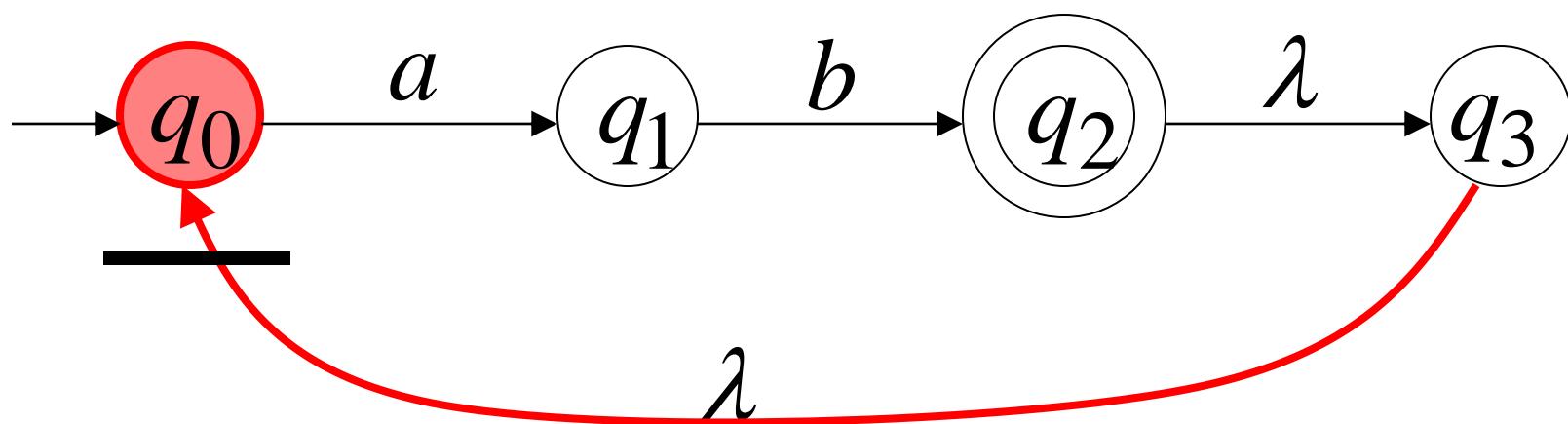
$a$	$b$	$a$	$b$	
-----	-----	-----	-----	--

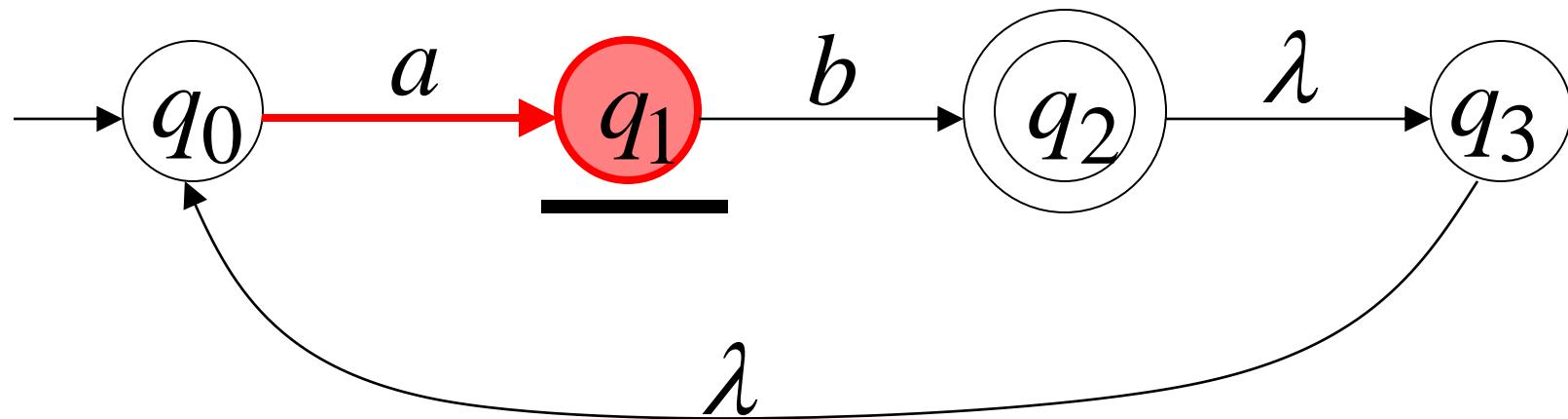
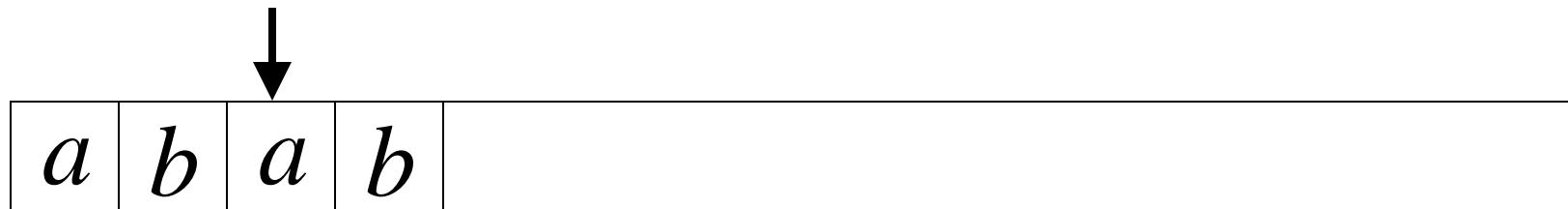


$a$	$b$	$a$	$b$	

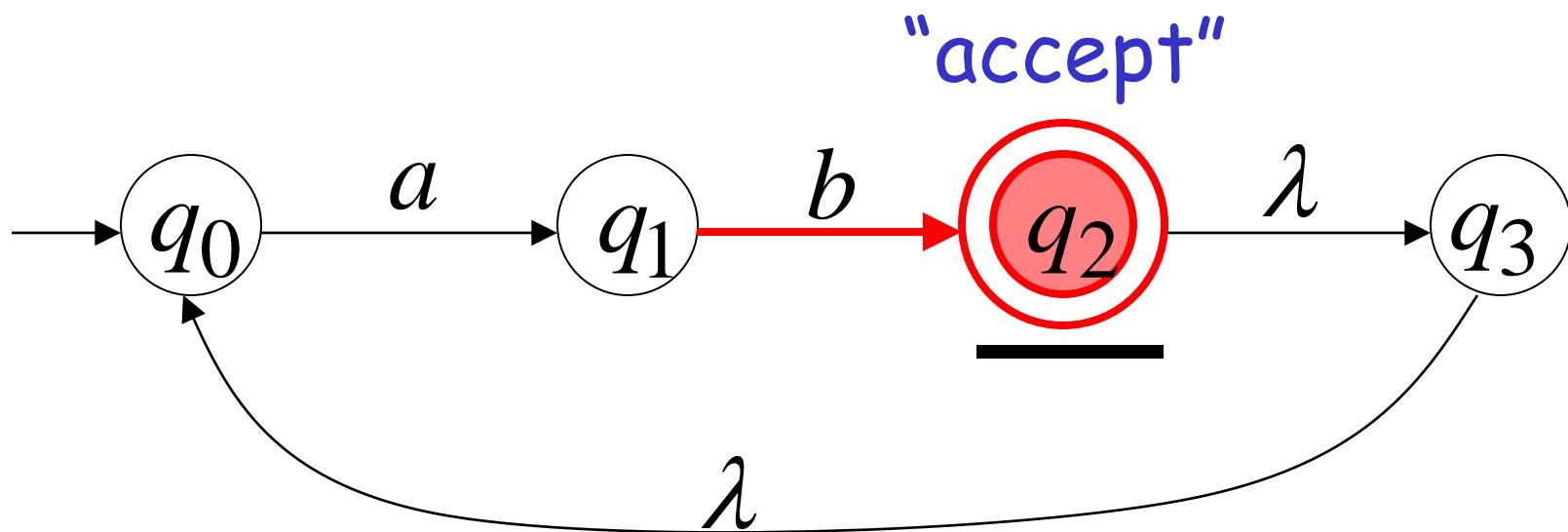


$a$	$b$	$a$	$b$	





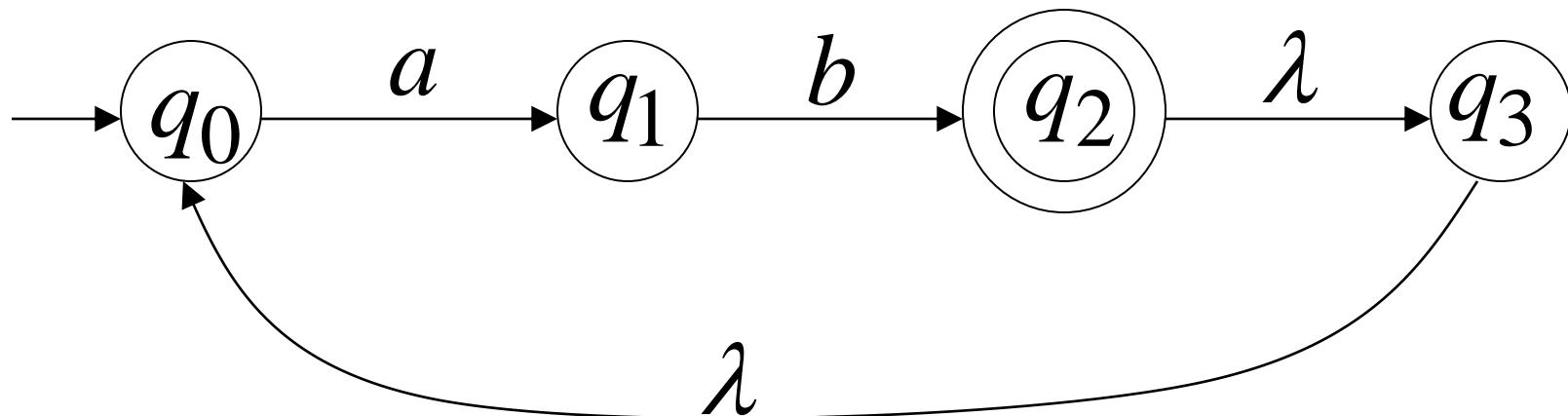
$a$	$b$	$a$	$b$	



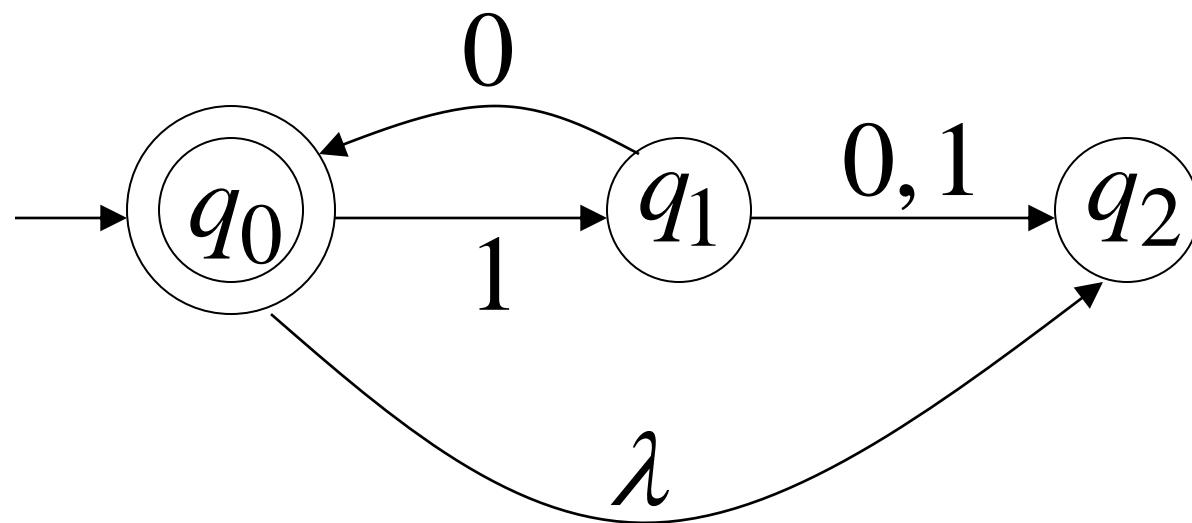
Language accepted

$$L = \{ab, abab, ababab, \dots\}$$

$$= \{ab\}^+$$

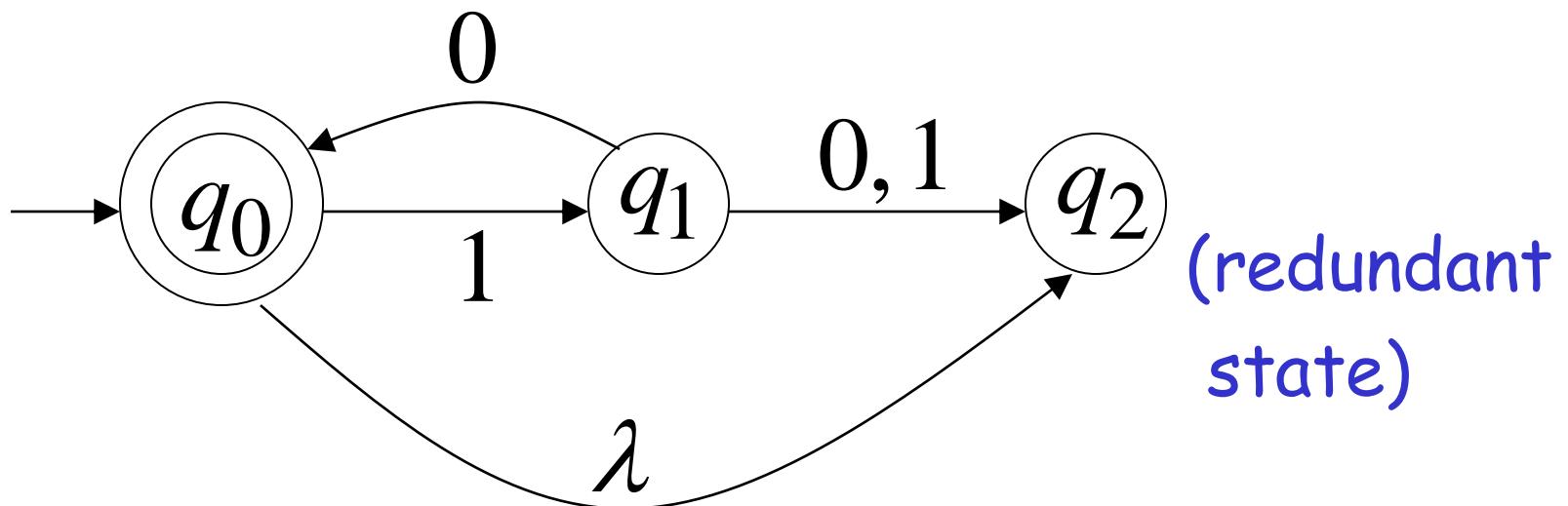


# Another NFA Example



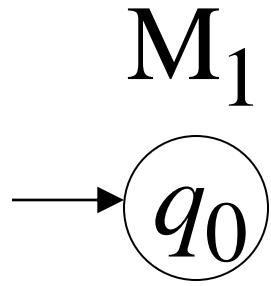
Language accepted

$$\begin{aligned}L(M) &= \{\lambda, 10, 1010, 101010, \dots\} \\&= \{10\}^*\end{aligned}$$

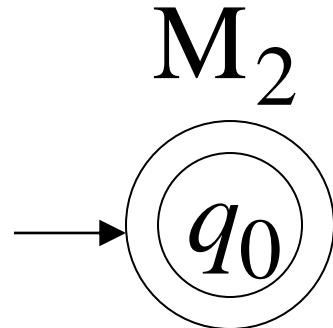


## Remarks:

- The  $\lambda$  symbol never appears on the input tape
- Simple automata:

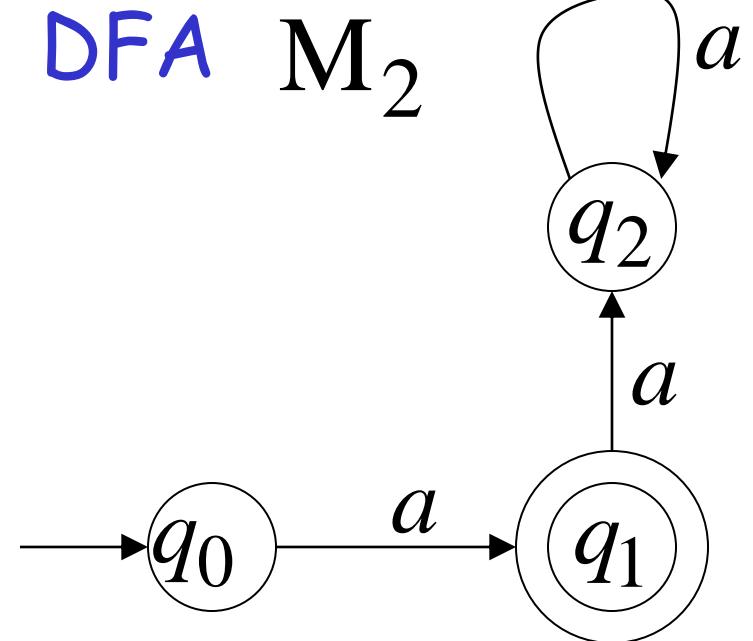
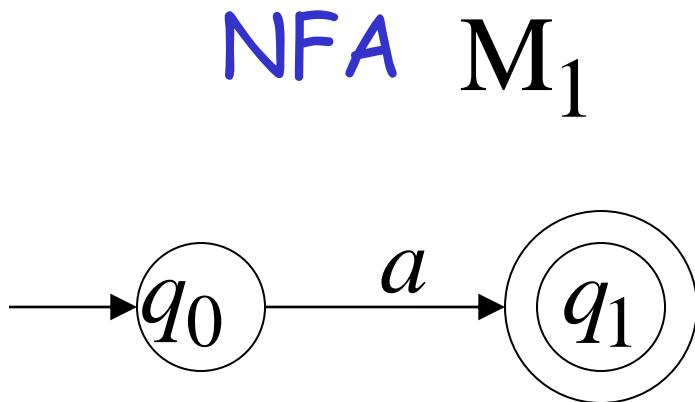


$$L(M_1) = \{ \}$$



$$L(M_2) = \{ \lambda \}$$

- NFAs are interesting because we can express languages easier than DFAs



$$L(M_1) = \{a\}$$

$$L(M_2) = \{a\}$$

# Formal Definition of NFAs

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

$Q$ : Set of states, i.e.  $\{q_0, q_1, q_2\}$

$\Sigma$ : Input alphabet, i.e.  $\{a, b\}$        $\lambda \notin \Sigma$

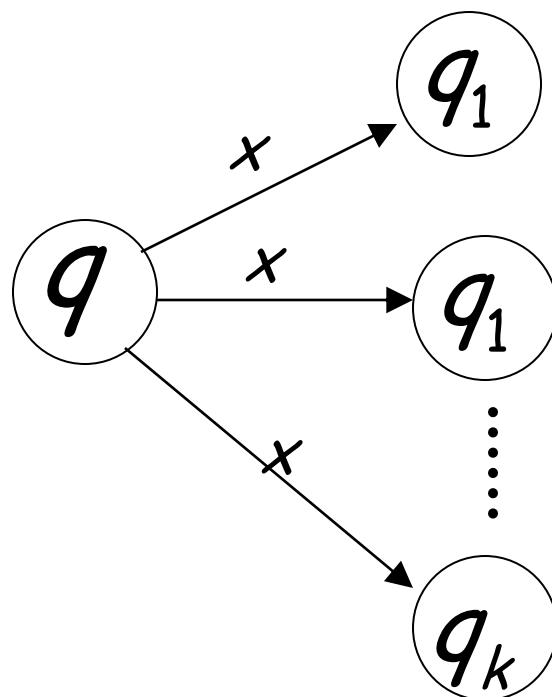
$\delta$ : Transition function

$q_0$ : Initial state

$F$ : Accepting states

# Transition Function $\delta$

$$\delta(q, x) = \{q_1, q_2, \dots, q_k\}$$



resulting states with  
following **one** transition  
with symbol  $x$

# Formal Notation - Epsilon Transition

- Transition function  $\delta$  is now a function that takes as arguments:
  - A state in  $Q$  and
  - A member of  $\Sigma \cup \{\epsilon\}$ ; that is, an input symbol or the symbol  $\epsilon$ . We require that  $\epsilon$  not be a symbol of the alphabet  $\Sigma$  to avoid any confusion.

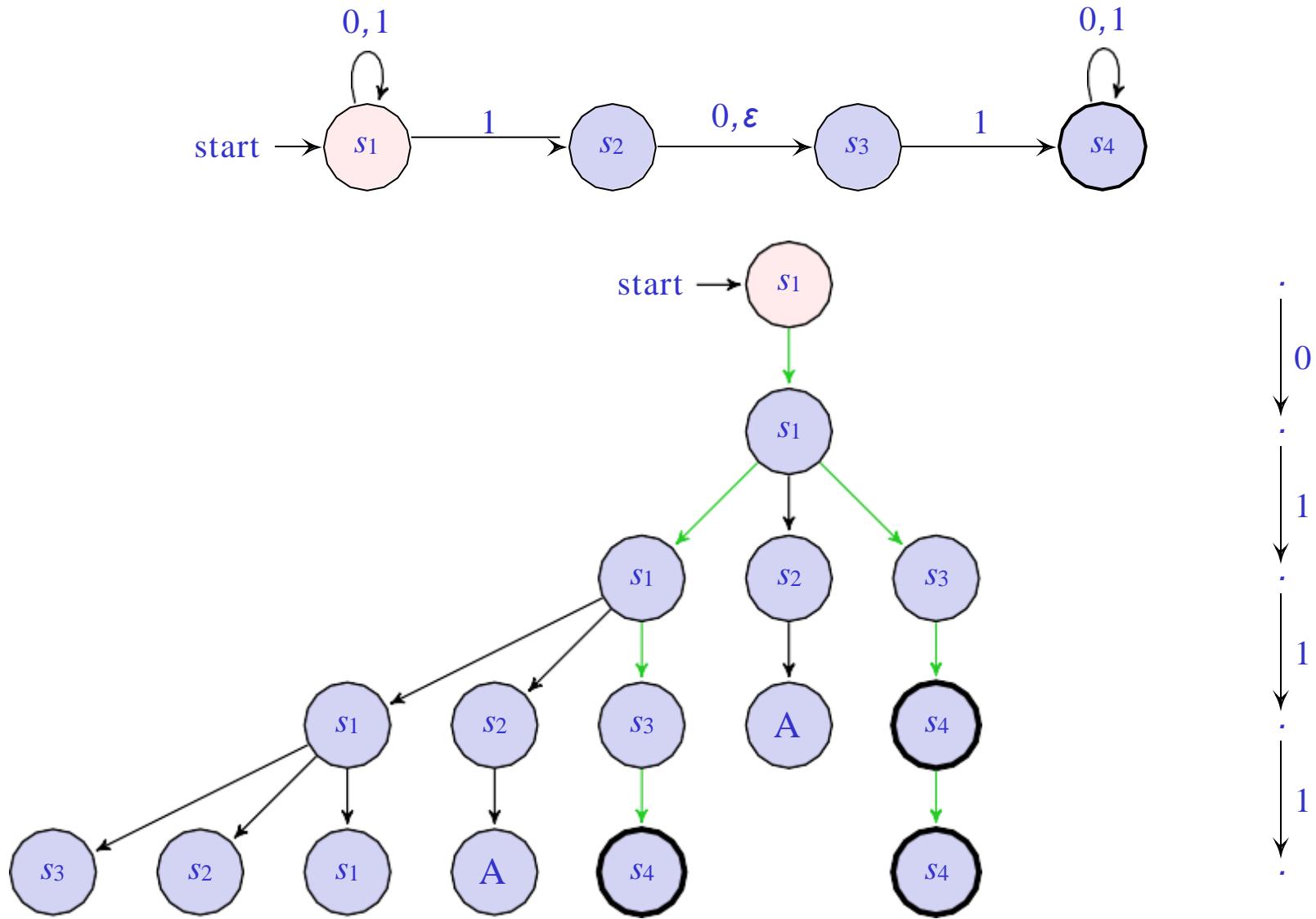
# NFA's With $\epsilon$ -Transitions

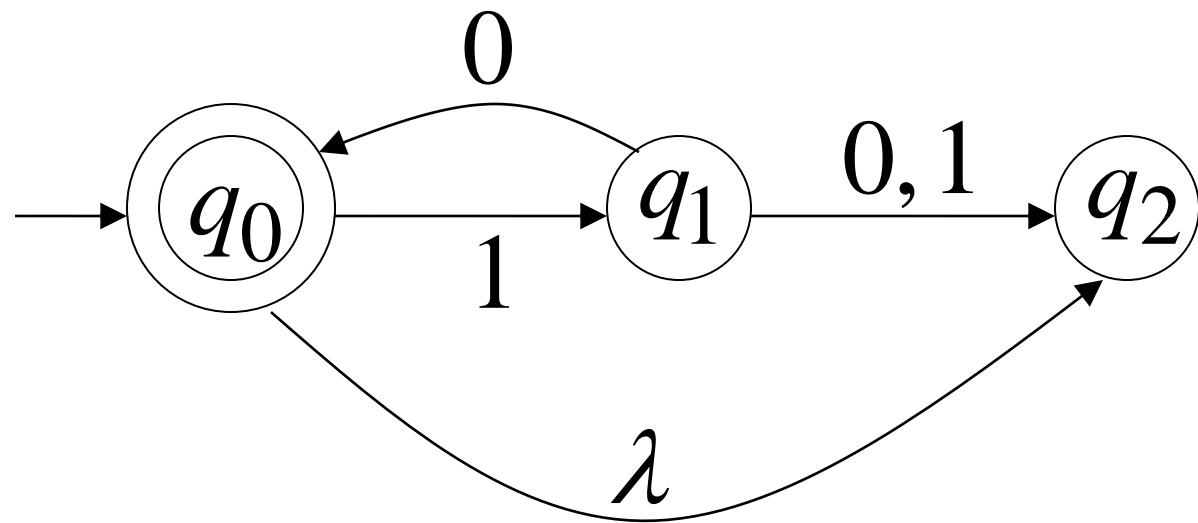
- We can allow state-to-state transitions on  $\epsilon$  input.
- These transitions are done spontaneously, without looking at the input string.
- A convenience at times, but still only regular languages are accepted.

# Corollary

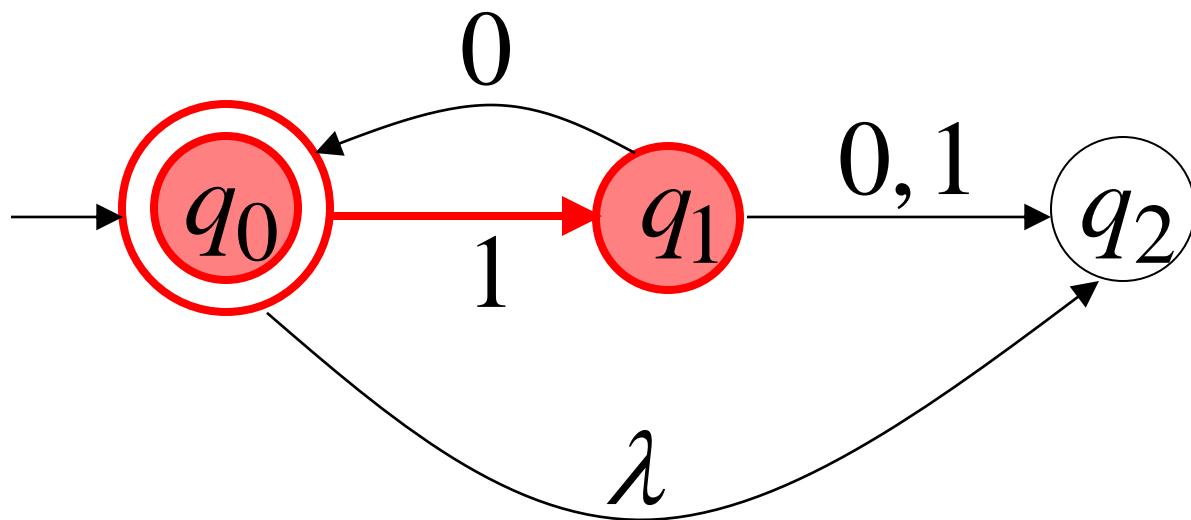
- A language is regular if and only if some nondeterministic finite automaton recognizes it
- A language is regular if and only if some deterministic finite automaton recognizes it

# Computation of an NFA

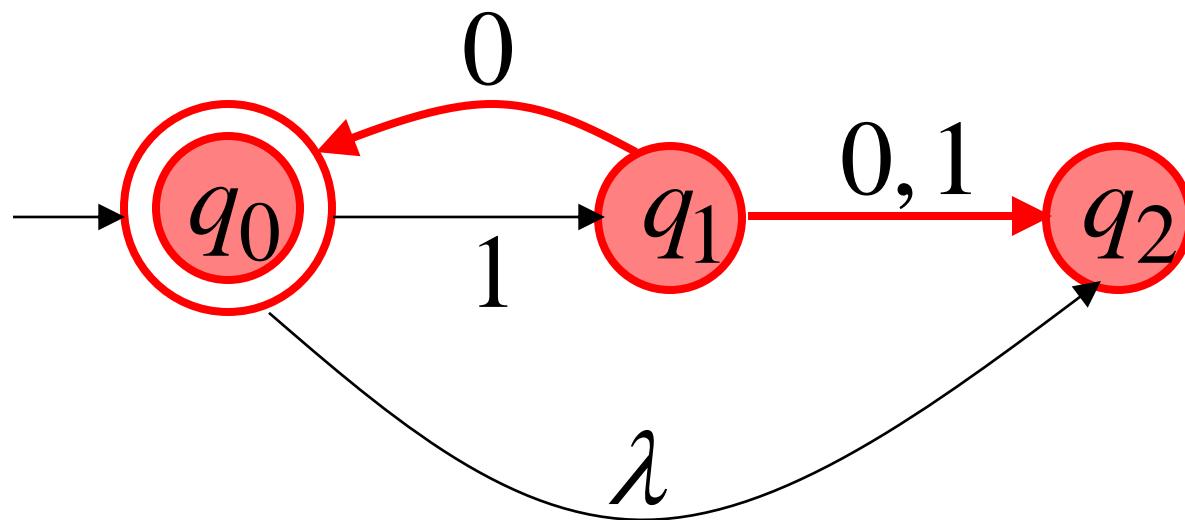




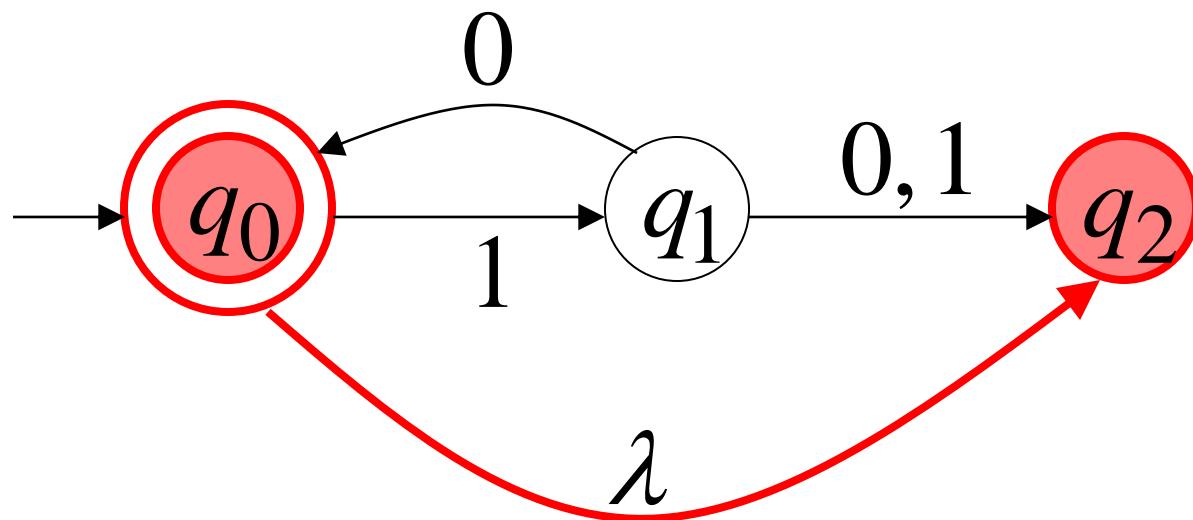
$$\delta(q_0, 1) = \{q_1\}$$



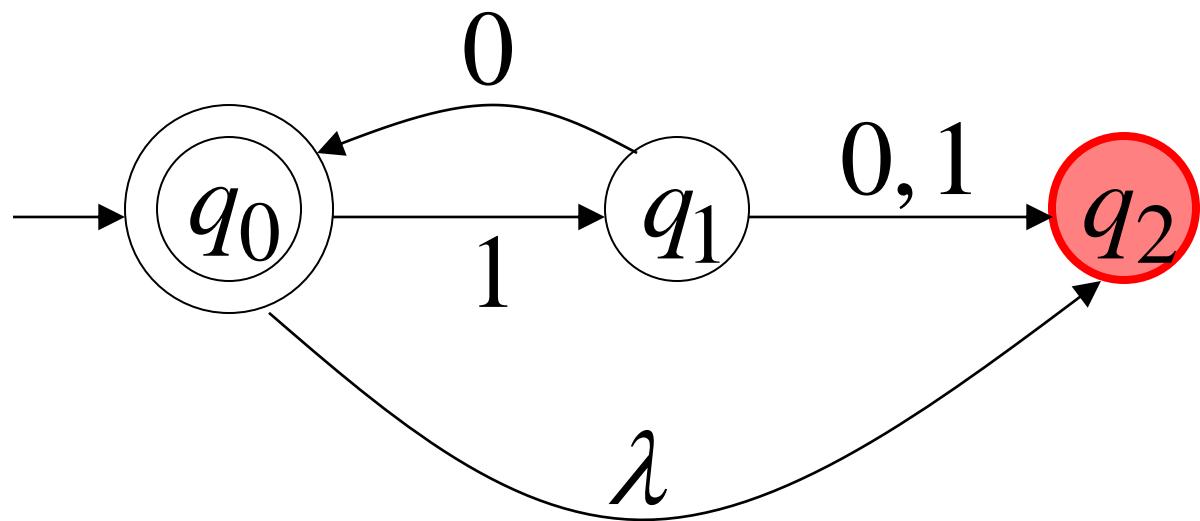
$$\delta(q_1, 0) = \{q_0, q_2\}$$



$$\delta(q_0, \lambda) = \{q_2\}$$



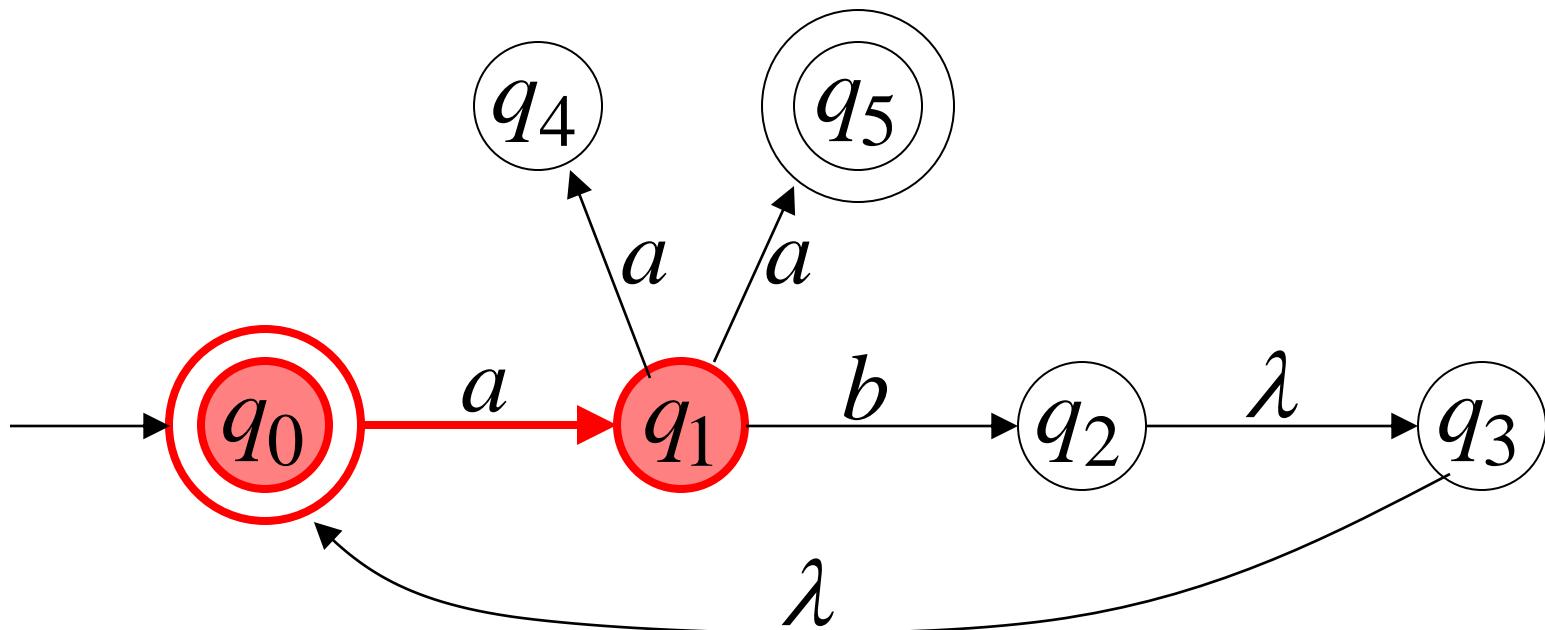
$$\delta(q_2, 1) = \emptyset$$



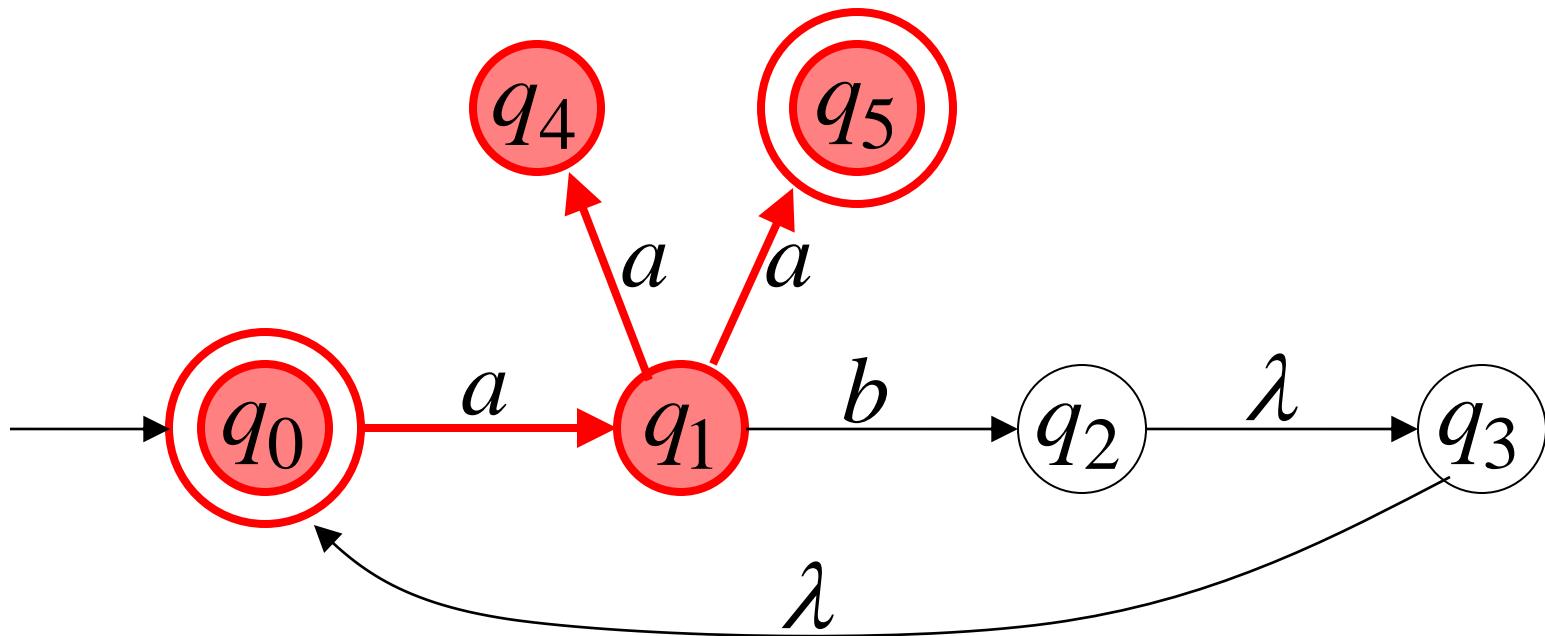
# Extended Transition Function $\delta^*$

Same with  $\delta$  but applied on strings

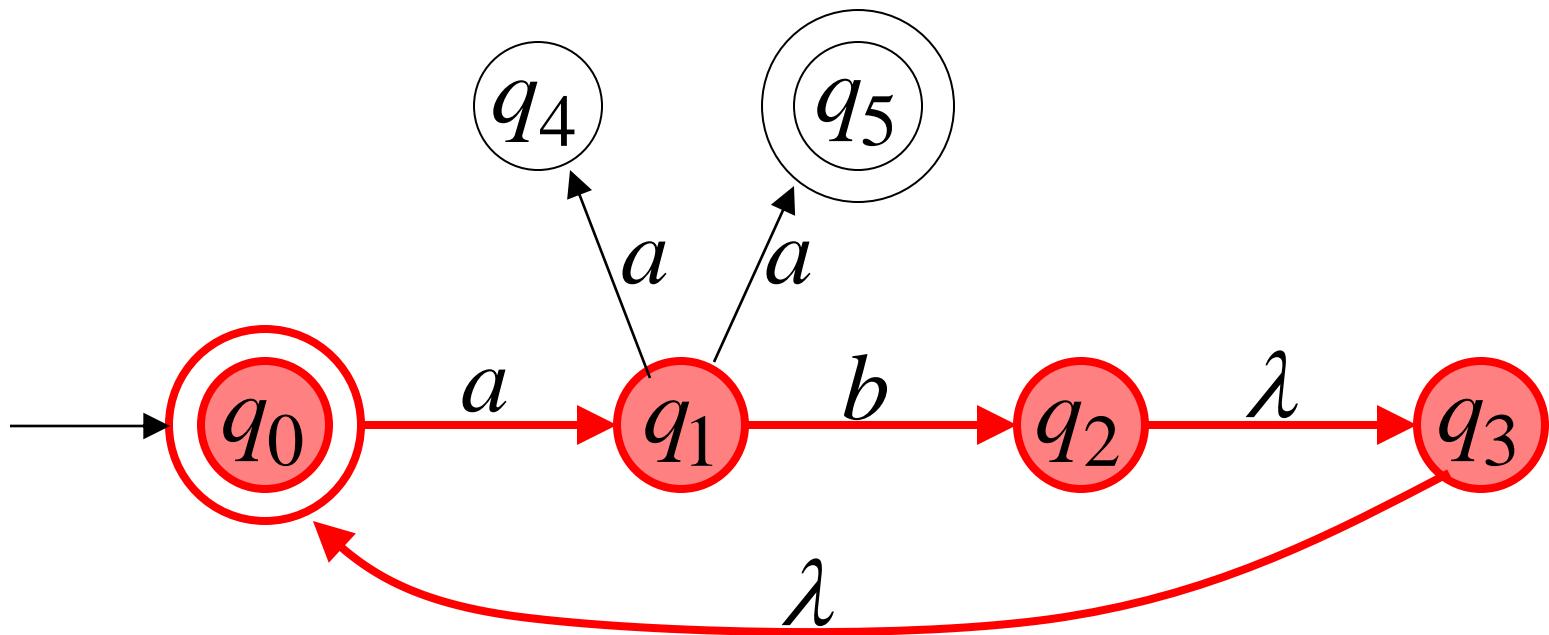
$$\delta^*(q_0, a) = \{q_1\}$$



$$\delta^*(q_0, aa) = \{q_4, q_5\}$$



$$\delta^*(q_0, ab) = \{q_2, q_3, q_0\}$$



Special case:

for any state  $q$

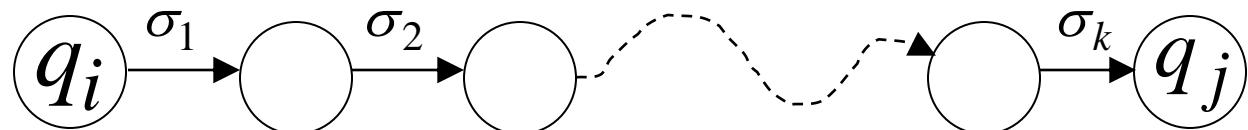
$$q \in \delta^*(q, \lambda)$$

In general

$q_j \in \delta^*(q_i, w)$  : there is a walk from  $q_i$  to  $q_j$   
with label  $w$



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



# The Language of an NFA $M$

The language accepted by  $M$  is:

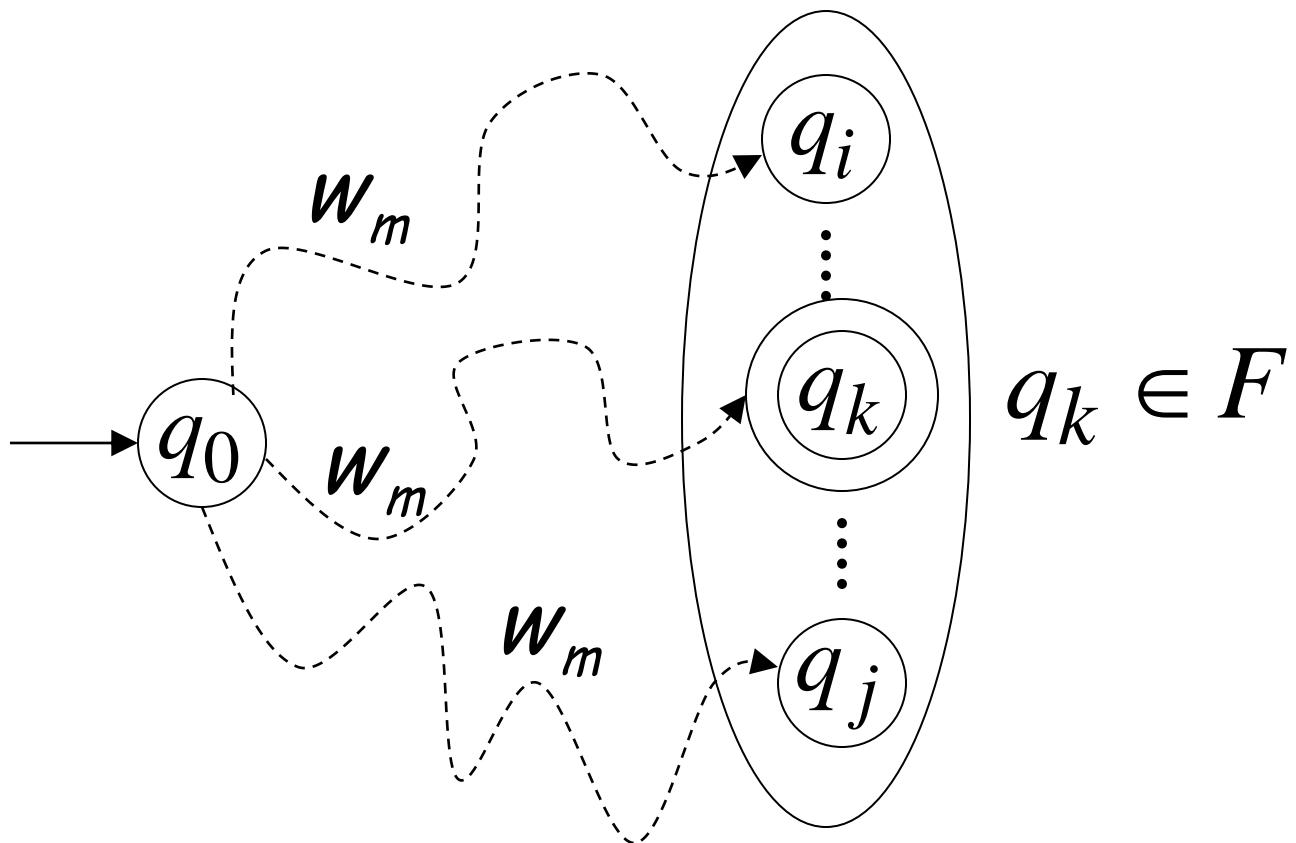
$$L(M) = \{w_1, w_2, \dots, w_n\}$$

where  $\delta^*(q_0, w_m) = \{q_i, \dots, q_k, \dots, q_j\}$

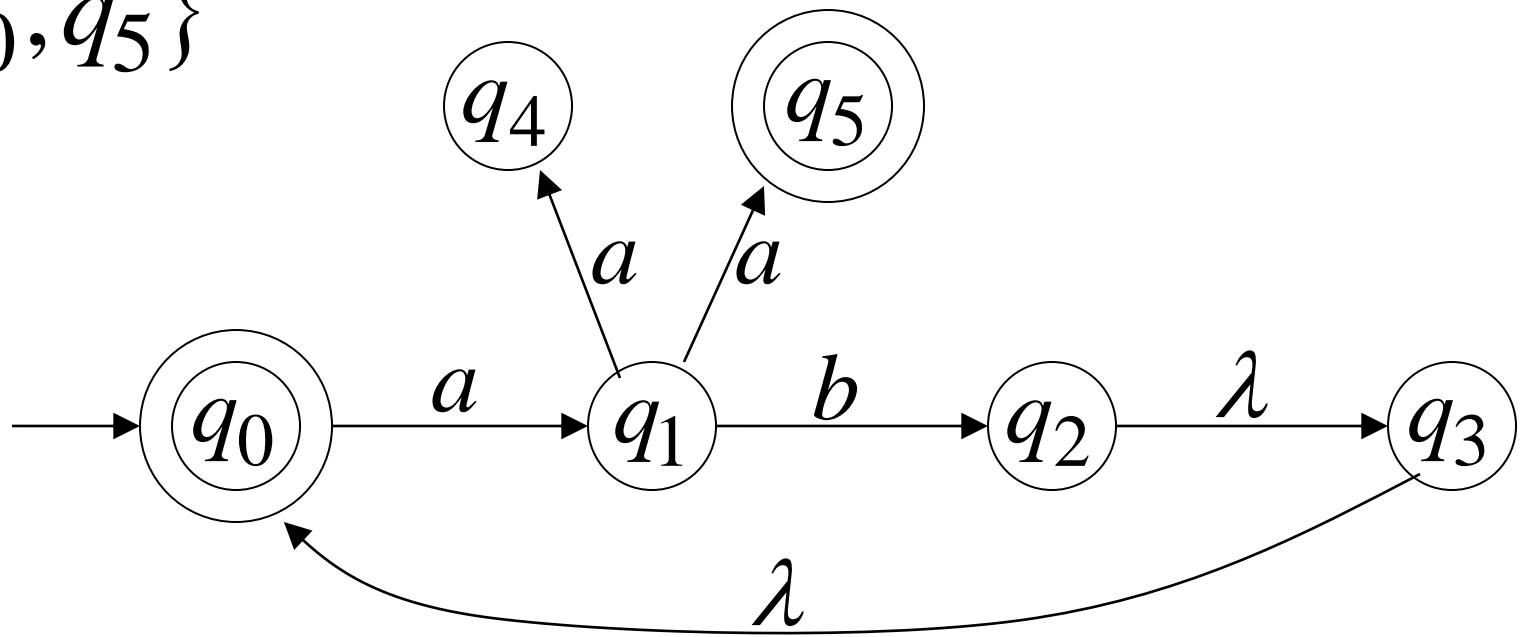
and there is some  $q_k \in F$  (accepting state)

$$w_m \in L(M)$$

$$\delta^*(q_0, w_m)$$



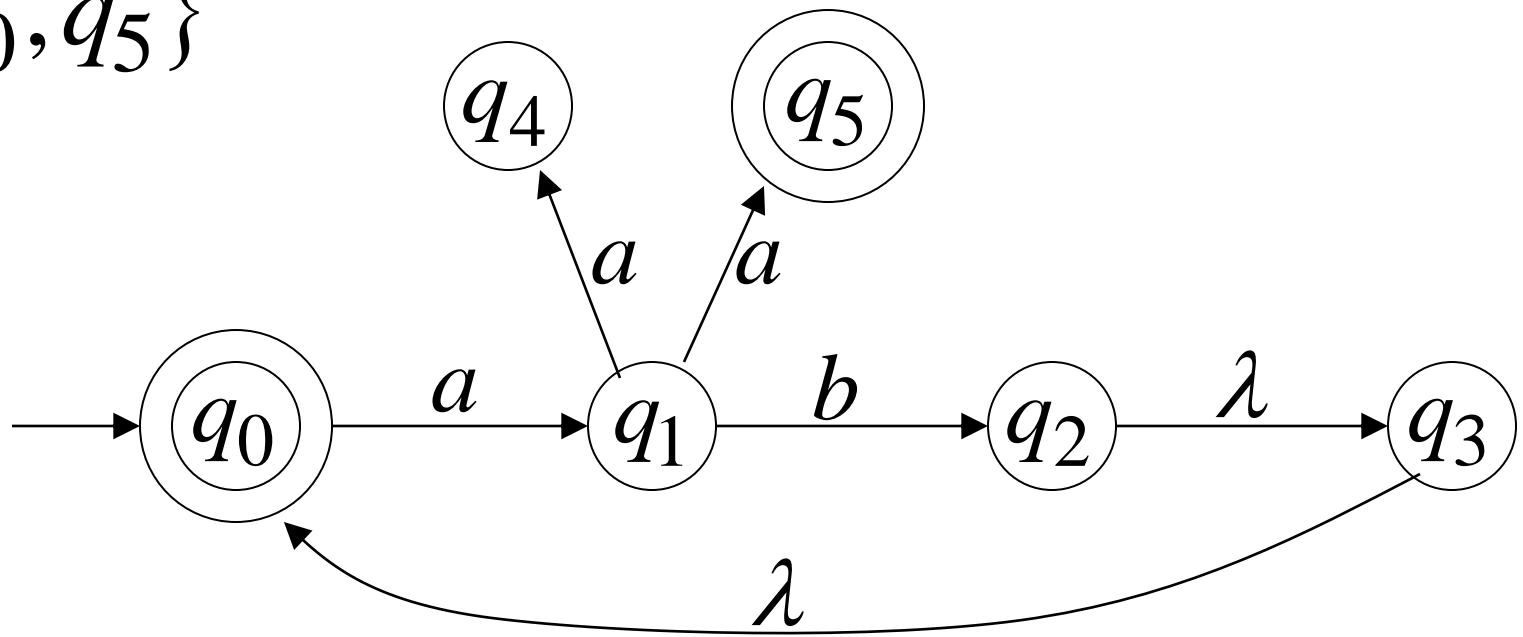
$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, aa) = \{q_4, \underline{q_5}\} \in F$$

$$aa \in L(M)$$

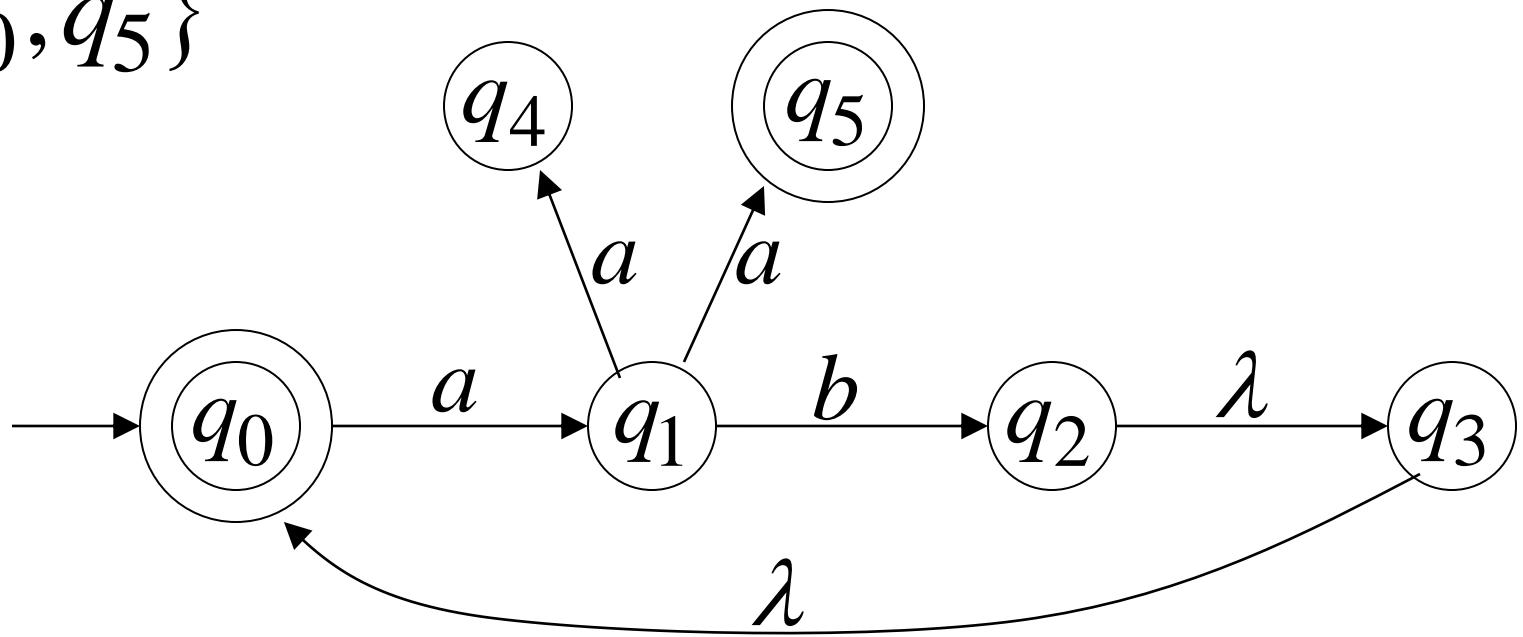
$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, ab) = \{q_2, q_3, \underline{q_0}\} \quad ab \in L(M)$$

$\in F$

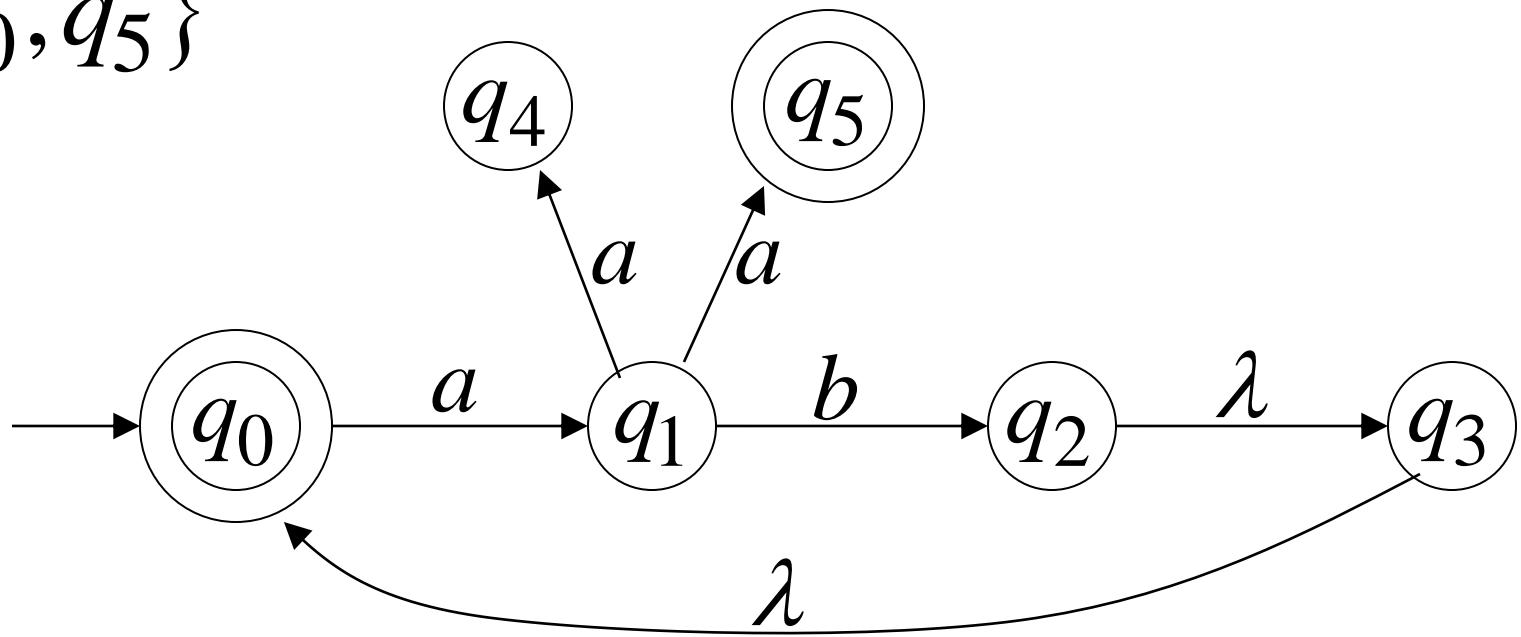
$$F = \{q_0, q_5\}$$



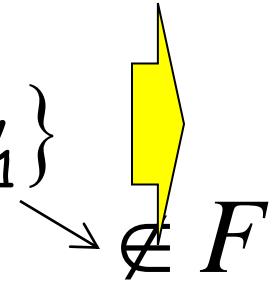
$$\delta^*(q_0, aba) = \{q_4, \underline{q_5}\} \xrightarrow{\quad \in F \quad}$$

$$aaba \in L(M)$$

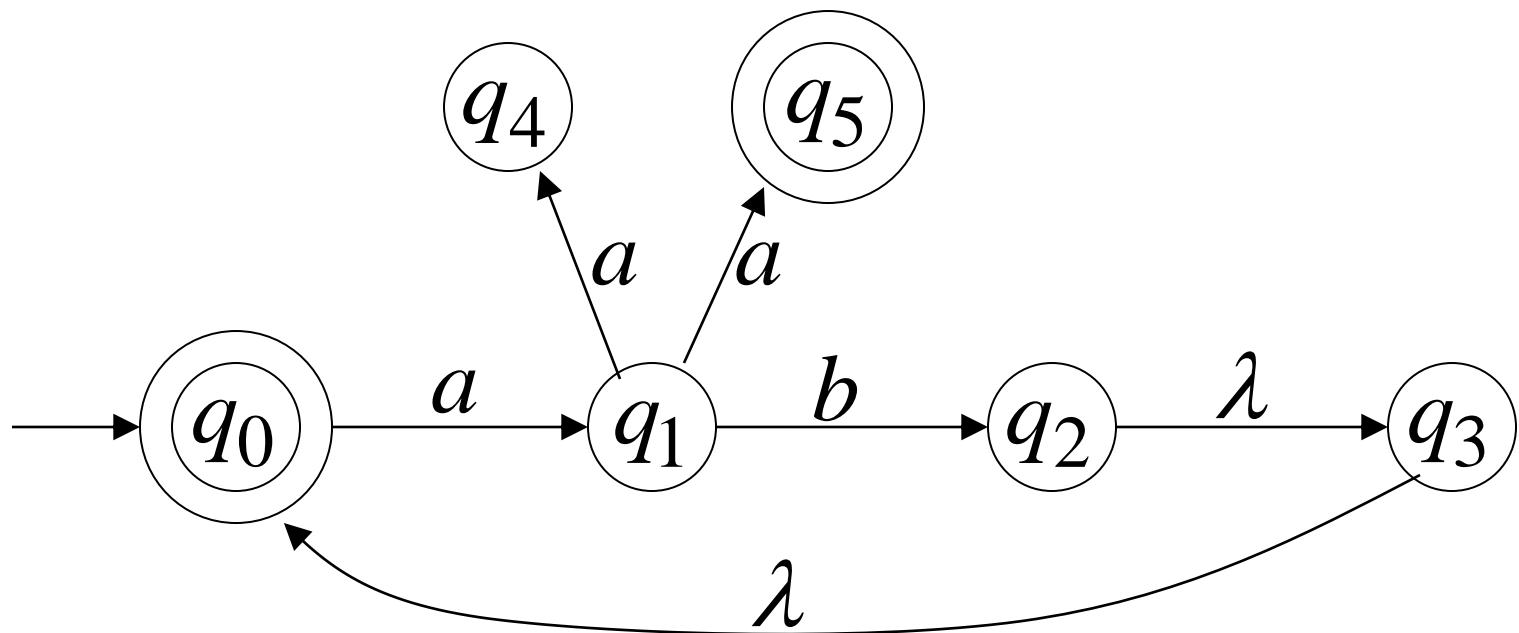
$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, aba) = \{q_1\}$$



$$aba \notin L(M)$$



$$L(M) = \{ab\}^* \cup \{ab\}^* \{aa\}$$

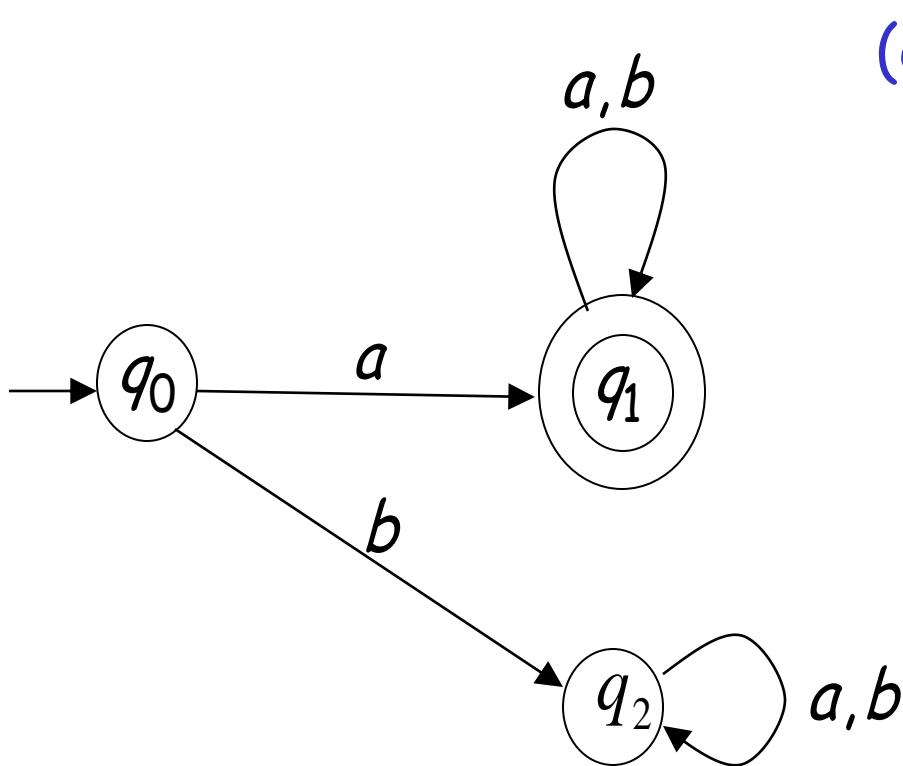
# Summary

- DFA's, NFA's, and  $\epsilon$ -NFA's all accept exactly the same set of languages: the regular languages.
- The NFA types are easier to design and may have exponentially fewer states than a DFA.

# DFA Minimization

# Minimization of FA

- Input - DFA
- Output - Minimized DFA
- Step 1 - Draw a table for all pairs of states  $(Q_i, Q_j)$  ( $i \neq j$ )  
[All are unmarked initially]

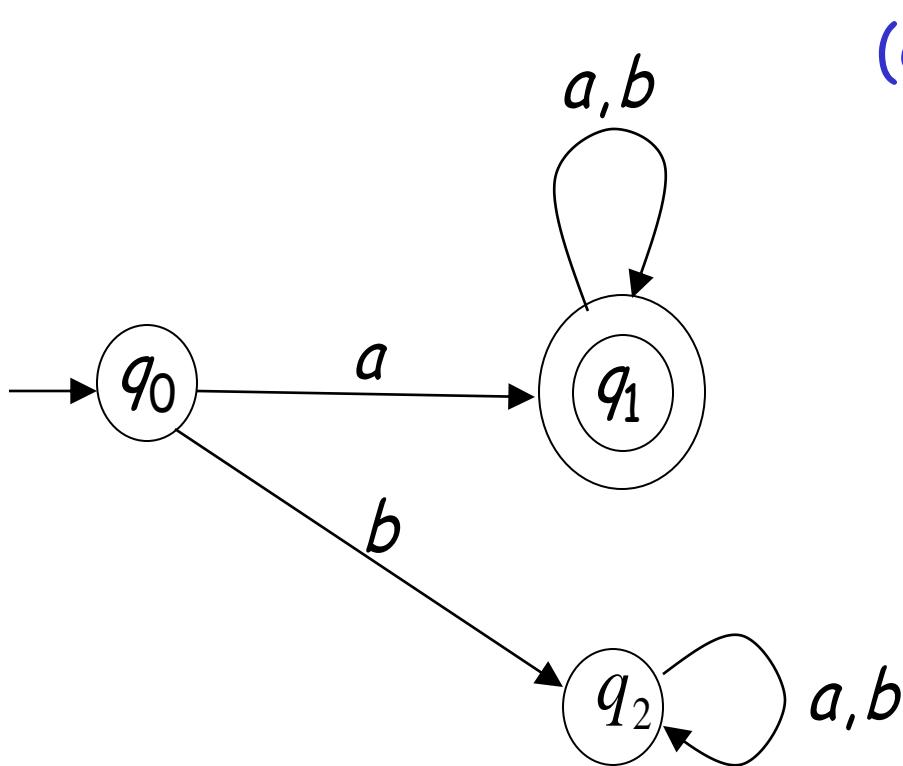


(q0,q1)    (q0,q2)    (q1,q2)

q0			
q1			
q2			
	q0	q1	q2

# Minimization of FA

- Input - DFA
- Output - Minimized DFA
- Step 1 - Draw a table for all pairs of states  $(Q_i, Q_j)$  ( $i \neq j$ )  
[All are unmarked initially]



(q0,q1)    (q0,q2)    (q1,q2)

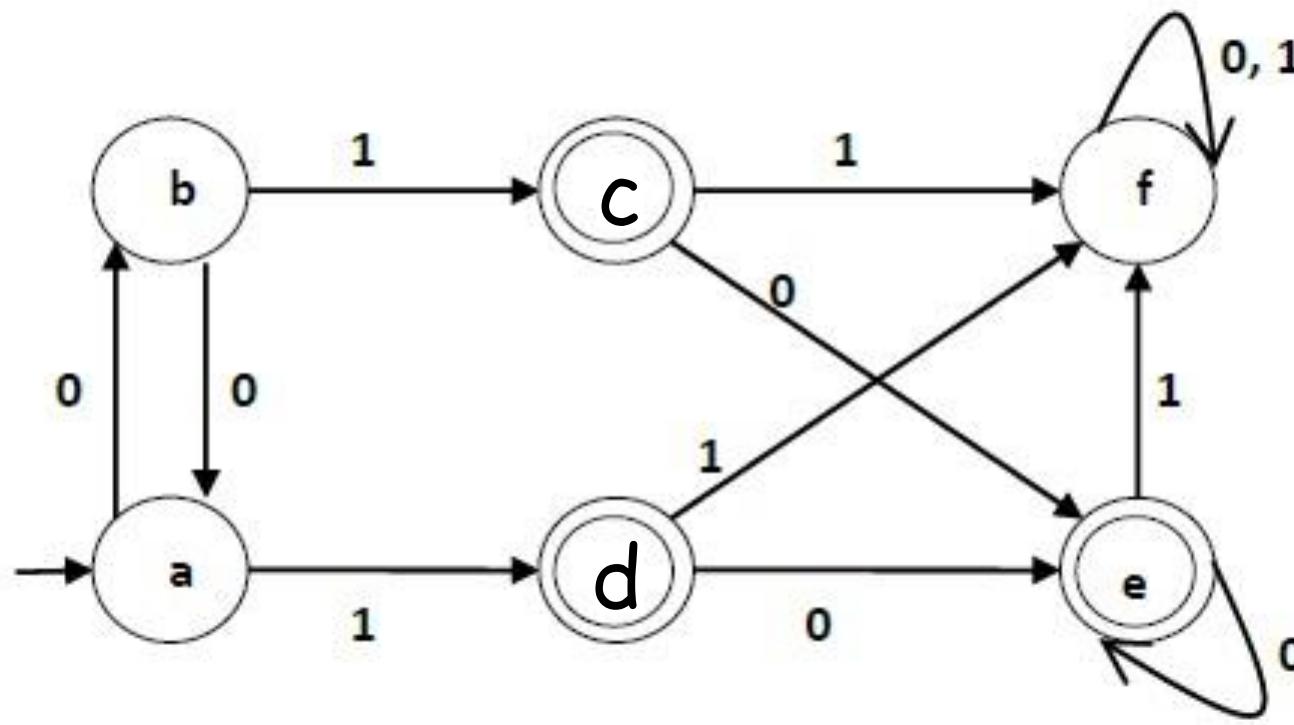
q1	
q2	
	q0    q1

# Minimization of FA

- **Input - DFA**
- **Output - Minimized DFA**
- **Step 1** - Draw a table for all pairs of states  $(Q_i, Q_j)$  ( $i \neq j$ )  
[All are unmarked initially]
- **Step 2** - Consider every state pair  $(Q_i, Q_j)$  in the DFA where  $Q_i \in F$  and  $Q_j \notin F$  or vice versa and mark them.  
[Here  $F$  is the set of final states]
- **Step 3** - Repeat this step until we cannot mark anymore states -  

If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.
- **Step 4** - Combine all the unmarked pair  $(Q_i, Q_j)$  and make them a single state in the reduced DFA.

# Example

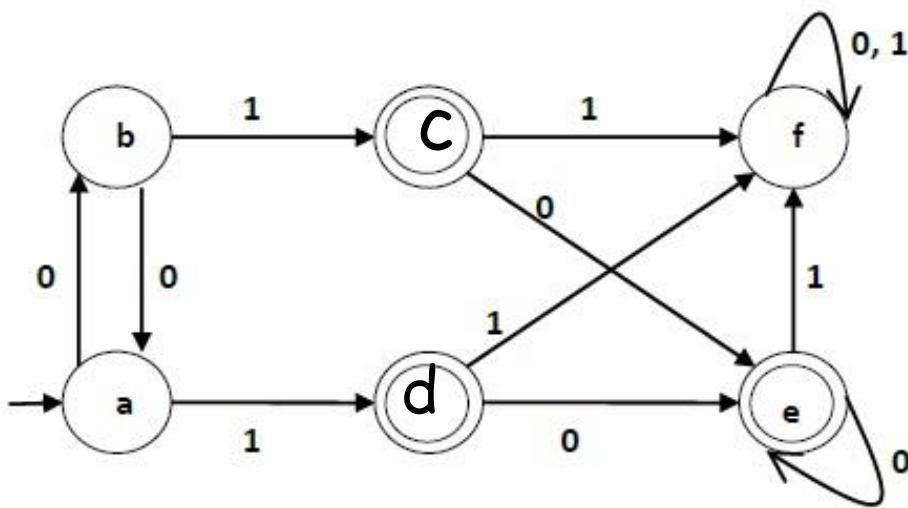


- Step 1 - Draw a table for all pairs of states ( $Q_i, Q_j$ ) [All are unmarked initially].

a						
b						
c						
d						
e						
f						
	a	b	c	d	e	F

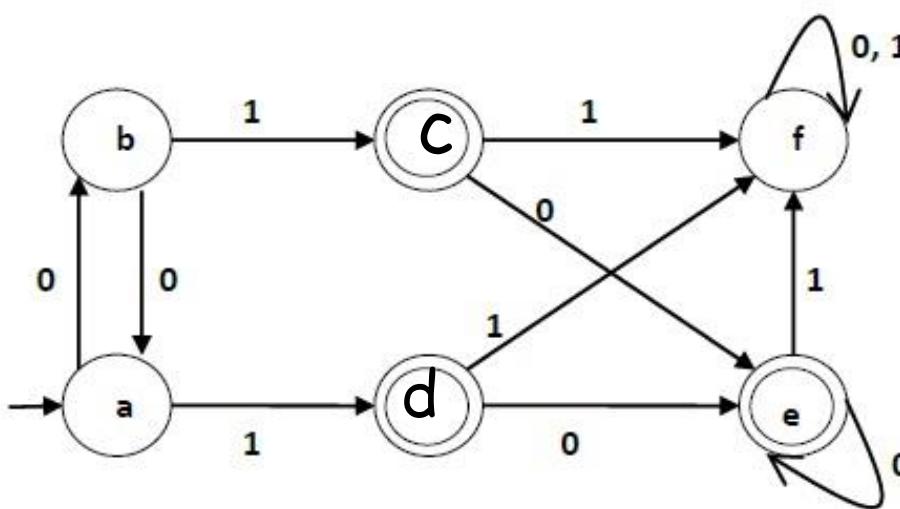
- Step 1 – Draw a table for all pairs of states ( $Q_i, Q_j$ ) [All are unmarked initially].

a						
b						
c						
d						
e						
f						
	a	b	c	d	e	F



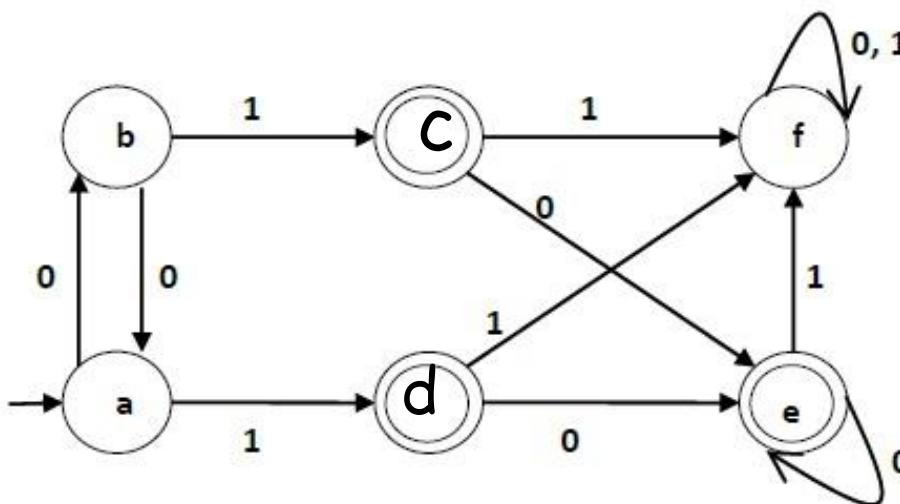
- Step 2 - Consider every state pair  $(Q_i, Q_j)$  in the DFA where  $Q_i \in F$  and  $Q_j \notin F$  or vice versa and mark them. [Here  $F$  is the set of final states] If both the states from pair are final then don't mark them.

a						
b						
c						
d						
e						
f						
	a	b	c	d	e	F



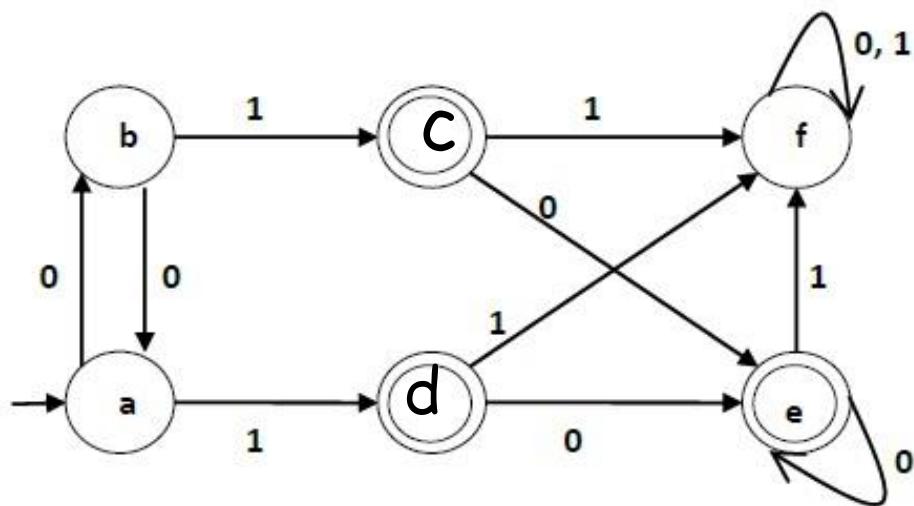
- Step 2 - Consider every state pair  $(Q_i, Q_j)$  in the DFA where  $Q_i \in F$  and  $Q_j \notin F$  or vice versa and mark them. [Here  $F$  is the set of final states]. If both the states from pair are final then don't mark them.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f			1	1	1	
	a	B	c	D	e	f



- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f			1	1	1	
	a	b	c	d	e	f



Unmarked pair is (a,f)

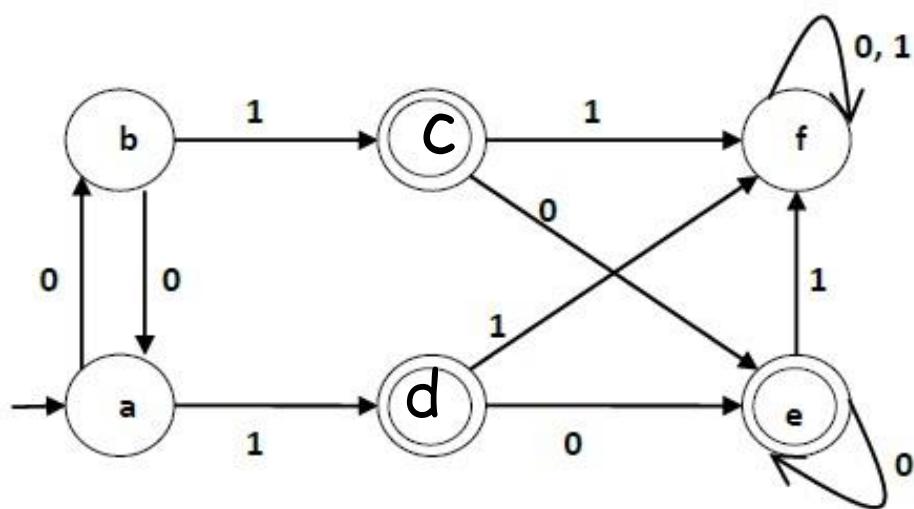
$$\begin{aligned}\delta(a, 0) &= b & (b, f) \\ \delta(f, 0) &= f\end{aligned}$$

Resultant pair are marked

$$\begin{aligned}\delta(a, 1) &= d & (d, f) \\ \delta(f, 1) &= f\end{aligned}$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2		1	1	1	
	a	b	c	d	e	f



Unmarked pair is (a,f)

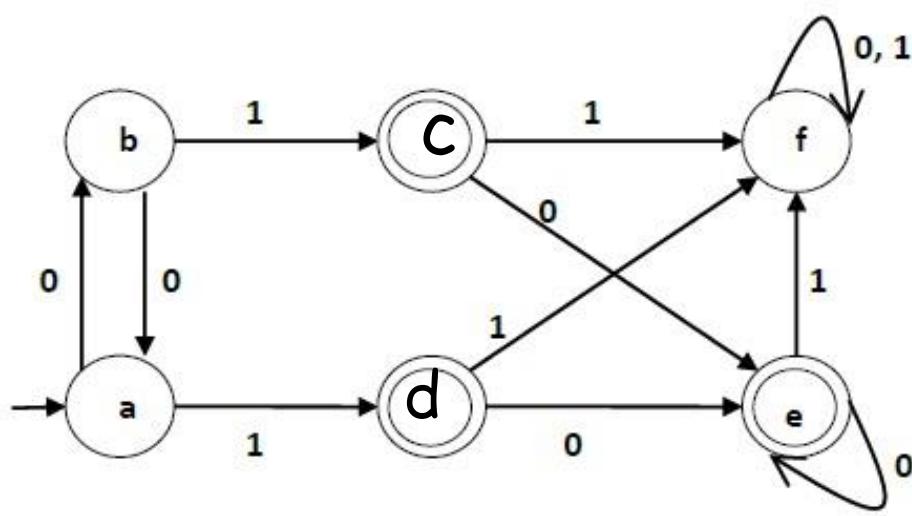
$$\begin{aligned}\delta(a, 0) &= b & (b, f) \\ \delta(f, 0) &= f\end{aligned}$$

Resultant pair are marked

$$\begin{aligned}\delta(a, 1) &= d & (d, f) \\ \delta(f, 1) &= f\end{aligned}$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2		1	1	1	
	a	b	c	d	e	f



Unmarked pair is (a,b)

$$\delta(a, 0) = b \quad (b, a)$$

$$\delta(b, 0) = a$$

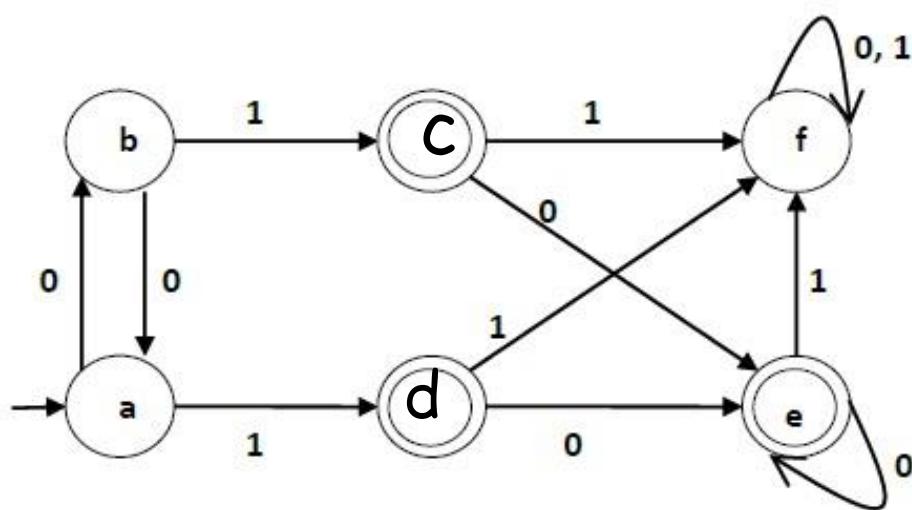
Resultant pair are unmarked

$$\delta(a, 1) = d \quad (d, c)$$

$$\delta(b, 1) = c$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2		1	1	1	
	a	b	c	d	e	f



Unmarked pair is  $(b, f)$

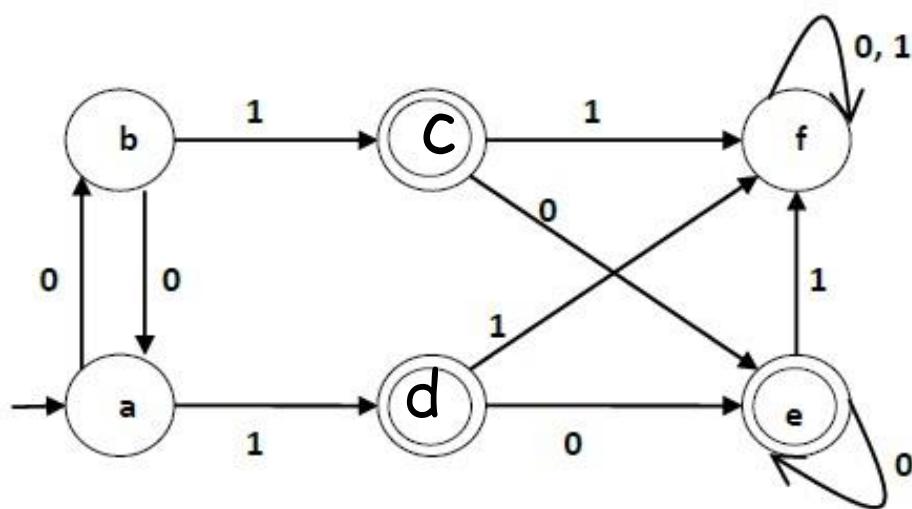
$$\begin{aligned}\delta(b, 0) &= a & (a, f) \\ \delta(f, 0) &= f\end{aligned}$$

Resultant pair are marked

$$\begin{aligned}\delta(b, 1) &= c & (c, f) \\ \delta(f, 1) &= f\end{aligned}$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2	2	1	1	1	
	a	b	c	d	e	f



Unmarked pair is  $(b, f)$

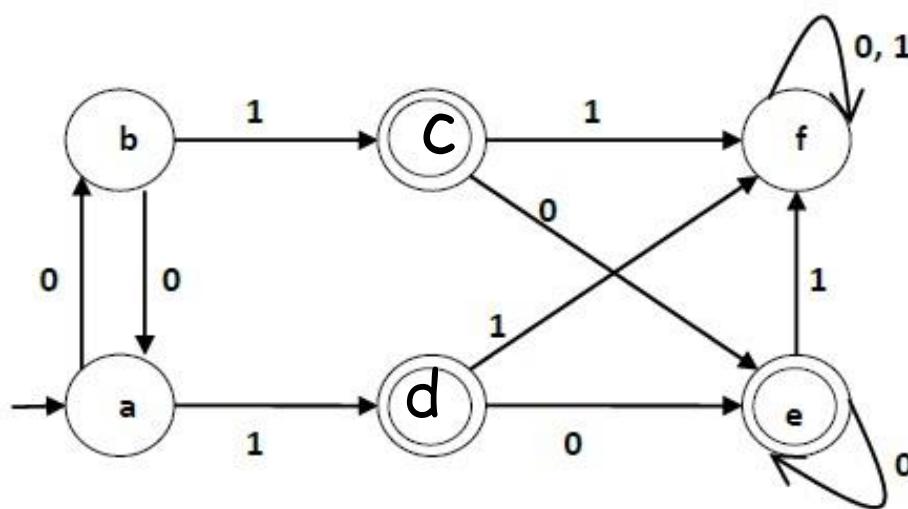
$$\begin{aligned}\delta(b, 0) &= a & (a, f) \\ \delta(f, 0) &= f\end{aligned}$$

Resultant pair are marked

$$\begin{aligned}\delta(b, 1) &= c & (c, f) \\ \delta(f, 1) &= f\end{aligned}$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2	2	1	1	1	
	a	b	c	d	e	f



Unmarked pair is  $(c, e)$

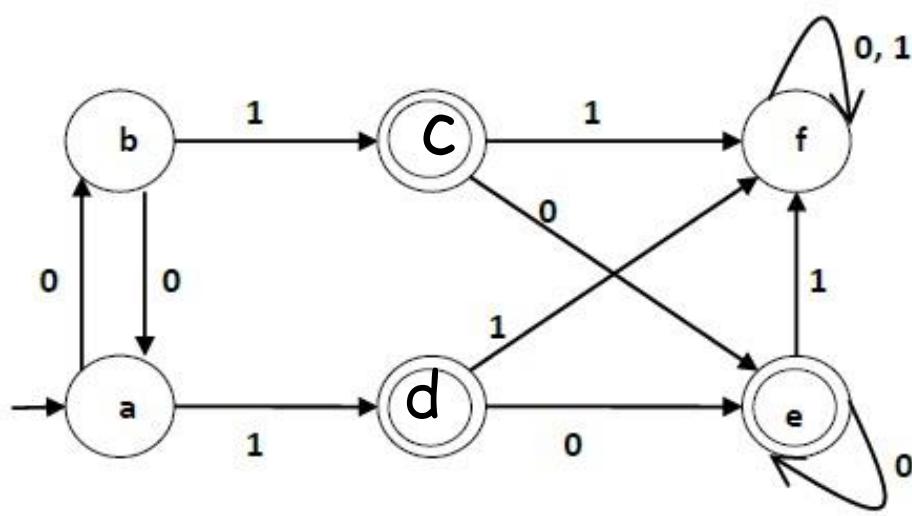
$$\begin{aligned}\delta(c, 0) &= e \\ \delta(e, 0) &= e\end{aligned}$$

Resultant pair are unmarked

$$\begin{aligned}\delta(c, 1) &= f \\ \delta(e, 1) &= f\end{aligned}$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2	2	1	1	1	
	a	b	c	d	e	f



Unmarked pair is  $(c,d)$

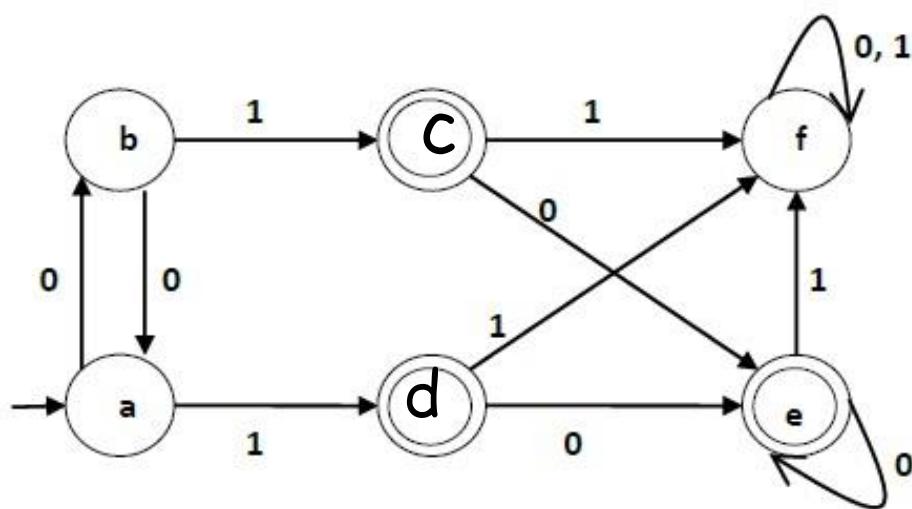
$$\begin{aligned}\delta(c, 0) &= e & (e, e) \\ \delta(d, 0) &= e\end{aligned}$$

Resultant pair are unmarked

$$\begin{aligned}\delta(c, 1) &= f & (f, f) \\ \delta(d, 1) &= f\end{aligned}$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2	2	1	1	1	
	a	b	c	d	e	f



Unmarked pair is  $(d, e)$

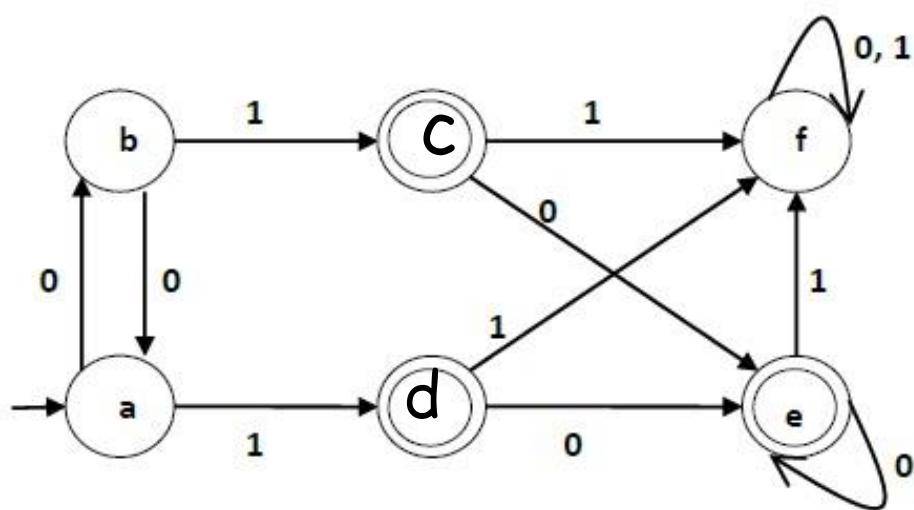
$$\begin{aligned}\delta(d, 0) &= e & (e, e) \\ \delta(e, 0) &= e\end{aligned}$$

Resultant pair are unmarked

$$\begin{aligned}\delta(d, 1) &= f & (f, f) \\ \delta(e, 1) &= f\end{aligned}$$

- Repeat this step until we cannot mark anymore states –  
If there is an unmarked pair  $(Q_i, Q_j)$ , mark it if the pair  $\{\delta(Q_i, A), \delta(Q_j, A)\}$  is marked for some input alphabet.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2	2	1	1	1	
	a	b	c	d	e	f



Unmarked pair is (a,b)

$$\delta(a, 0) = b \quad (b, a)$$

$$\delta(b, 0) = a$$

Resultant pair are unmarked

$$\delta(a, 1) = d \quad (d, c)$$

$$\delta(b, 1) = c$$

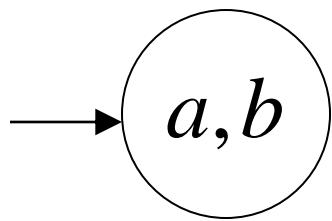
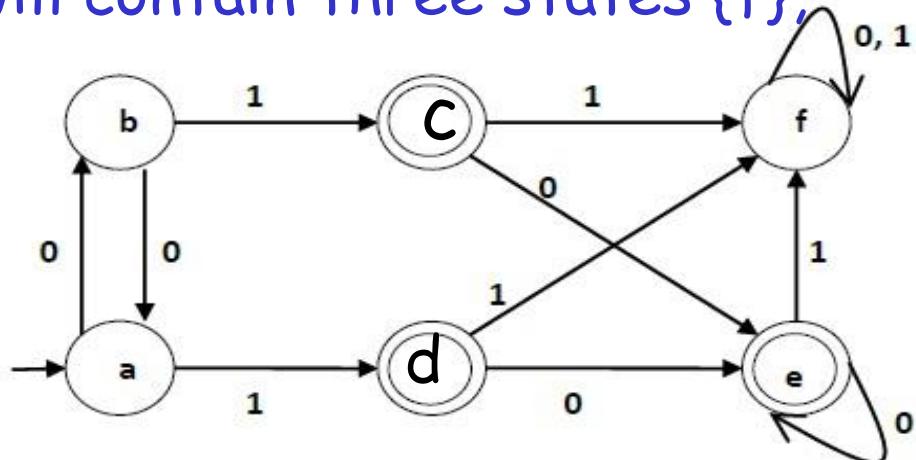
- Step 4 - Combine all the unmarked pair ( $Q_i, Q_j$ ) and make them a single state in the reduced DFA.

a						
b						
c	1	1				
d	1	1				
e	1	1				
f	2	2	1	1	1	
	a	b	c	d	e	f

- After step 3, we have got state combinations  $\{a, b\} \{c, d\} \{c, e\} \{d, e\}$  that are unmarked.
- We can recombine  $\{c, d\} \{c, e\} \{d, e\}$  into  $\{c, d, e\}$
- Hence we got two combined states as -  $\{a, b\}$  and  $\{c, d, e\}$

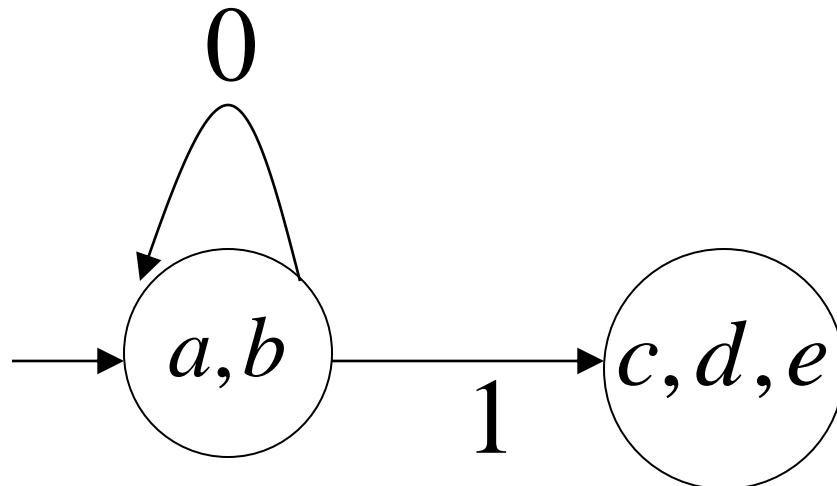
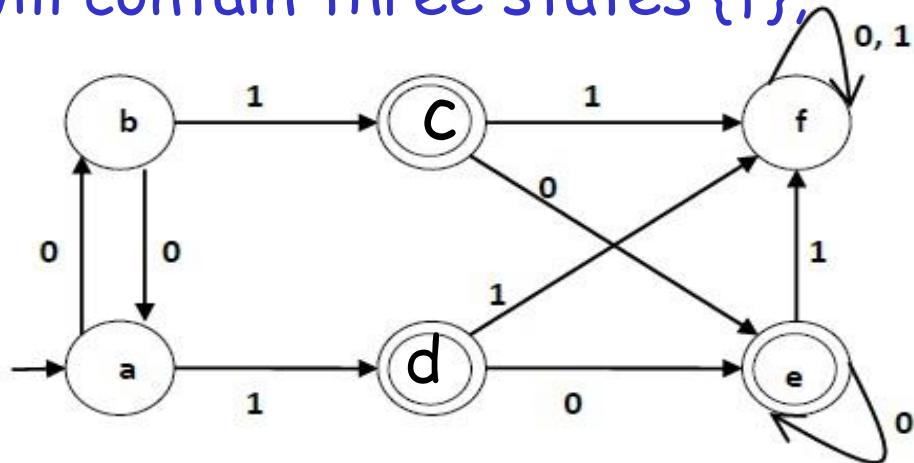
- So the final minimized DFA will contain three states  $\{f\}$ ,  $\{a, b\}$  and  $\{c, d, e\}$

$\{a, b\}$



- So the final minimized DFA will contain three states  $\{f\}$ ,  $\{a, b\}$  and  $\{c, d, e\}$
- $\{a, b\}$

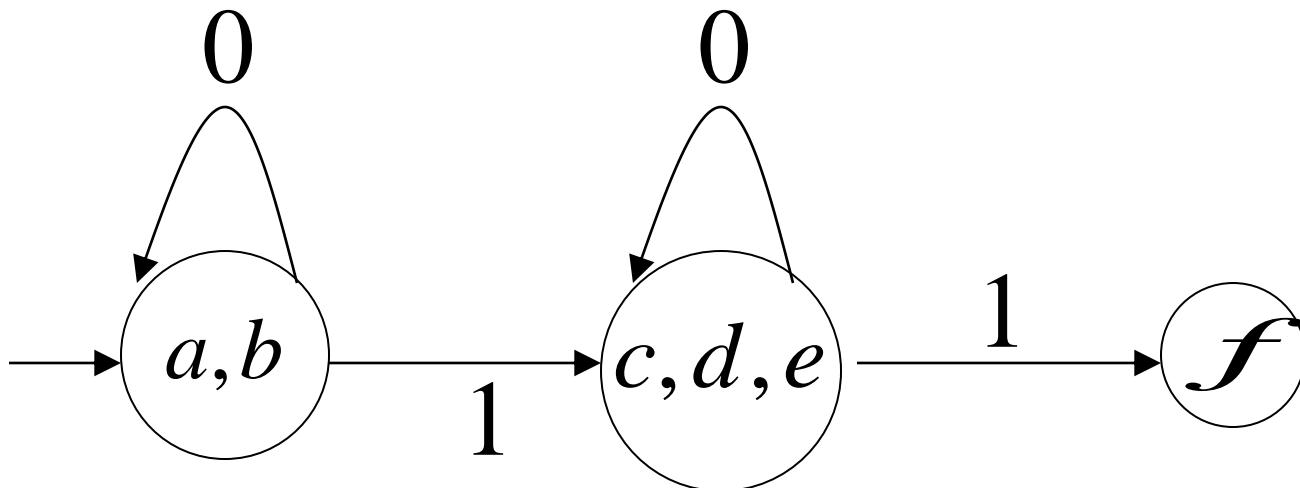
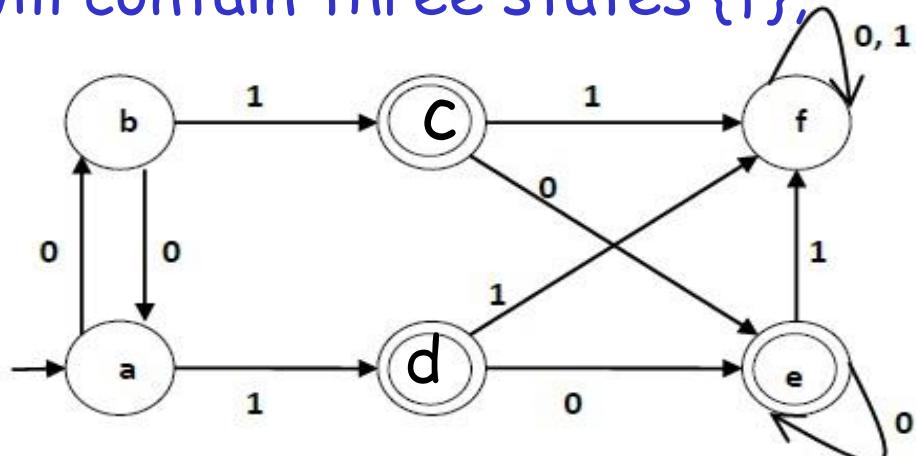
$$\begin{array}{ll} \delta(a, 0) = b & \delta(a, 1) = d \\ \delta(b, 0) = a & \delta(b, 1) = c \end{array}$$



- So the final minimized DFA will contain three states  $\{f\}$ ,  $\{a, b\}$  and  $\{c, d, e\}$

$\{c, d, e\}$

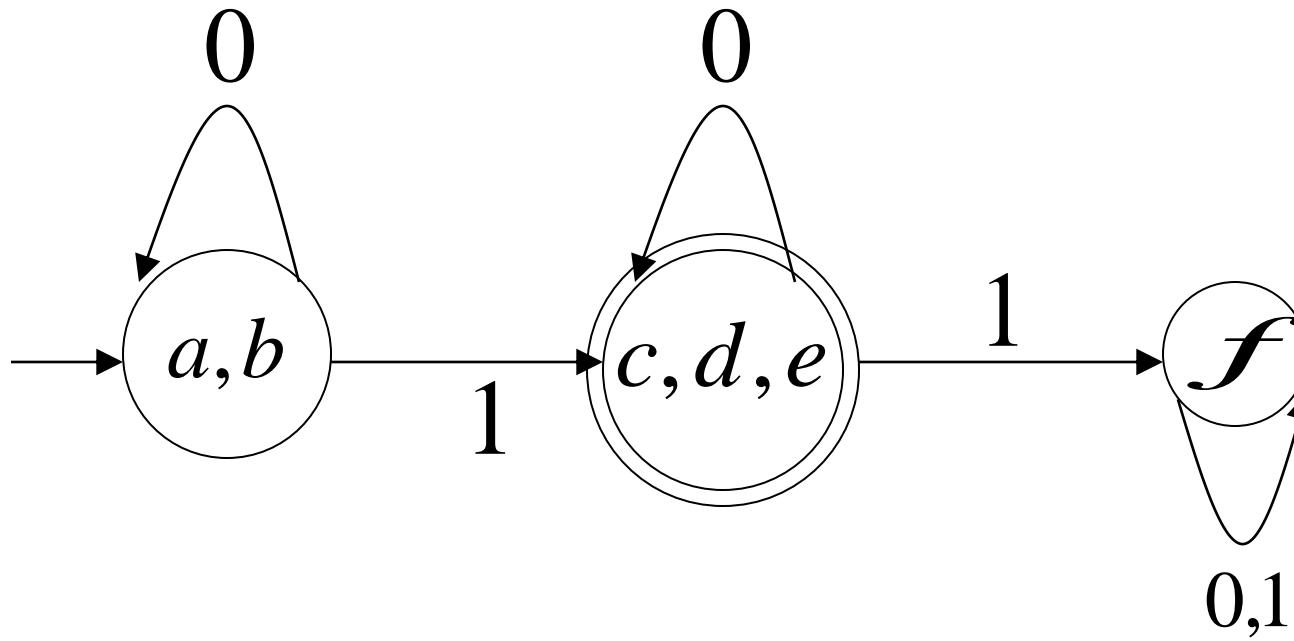
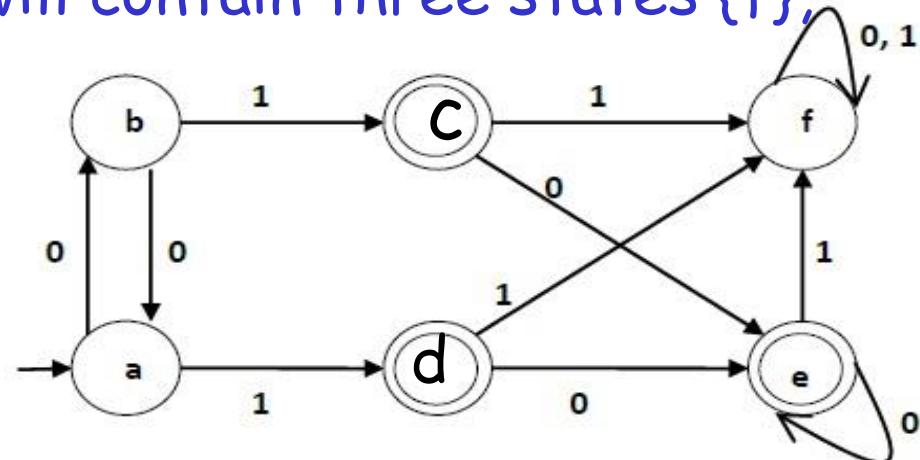
$$\begin{array}{ll} \delta(c, 0) = e & \delta(c, 1) = f \\ \delta(d, 0) = e & \delta(d, 1) = f \\ \delta(e, 0) = e & \delta(e, 1) = f \end{array}$$



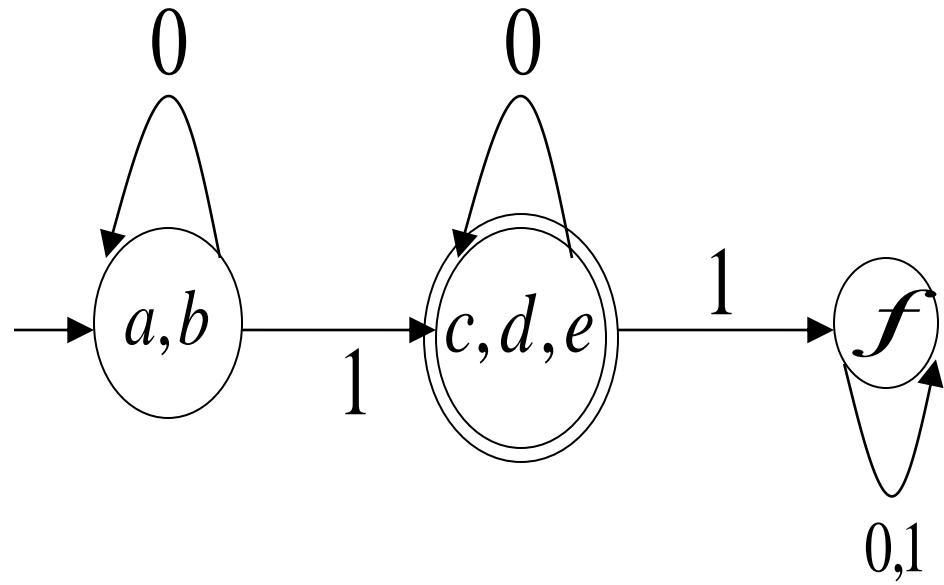
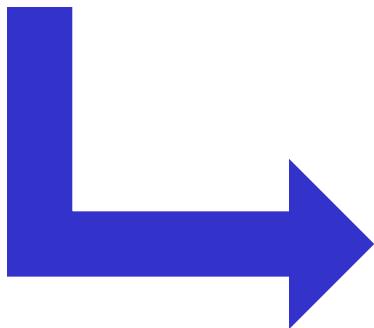
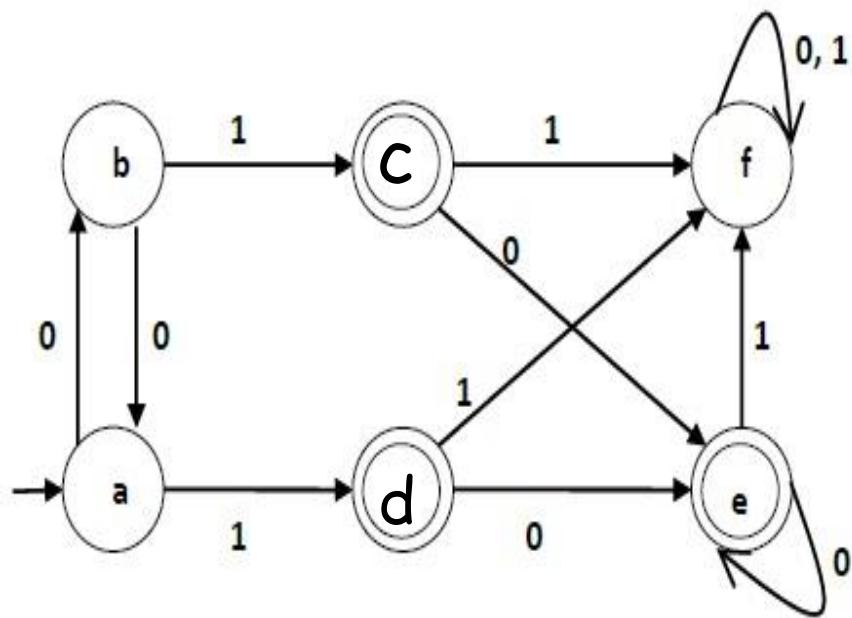
- So the final minimized DFA will contain three states  $\{f\}$ ,  $\{a, b\}$  and  $\{c, d, e\}$

$\{f\}$

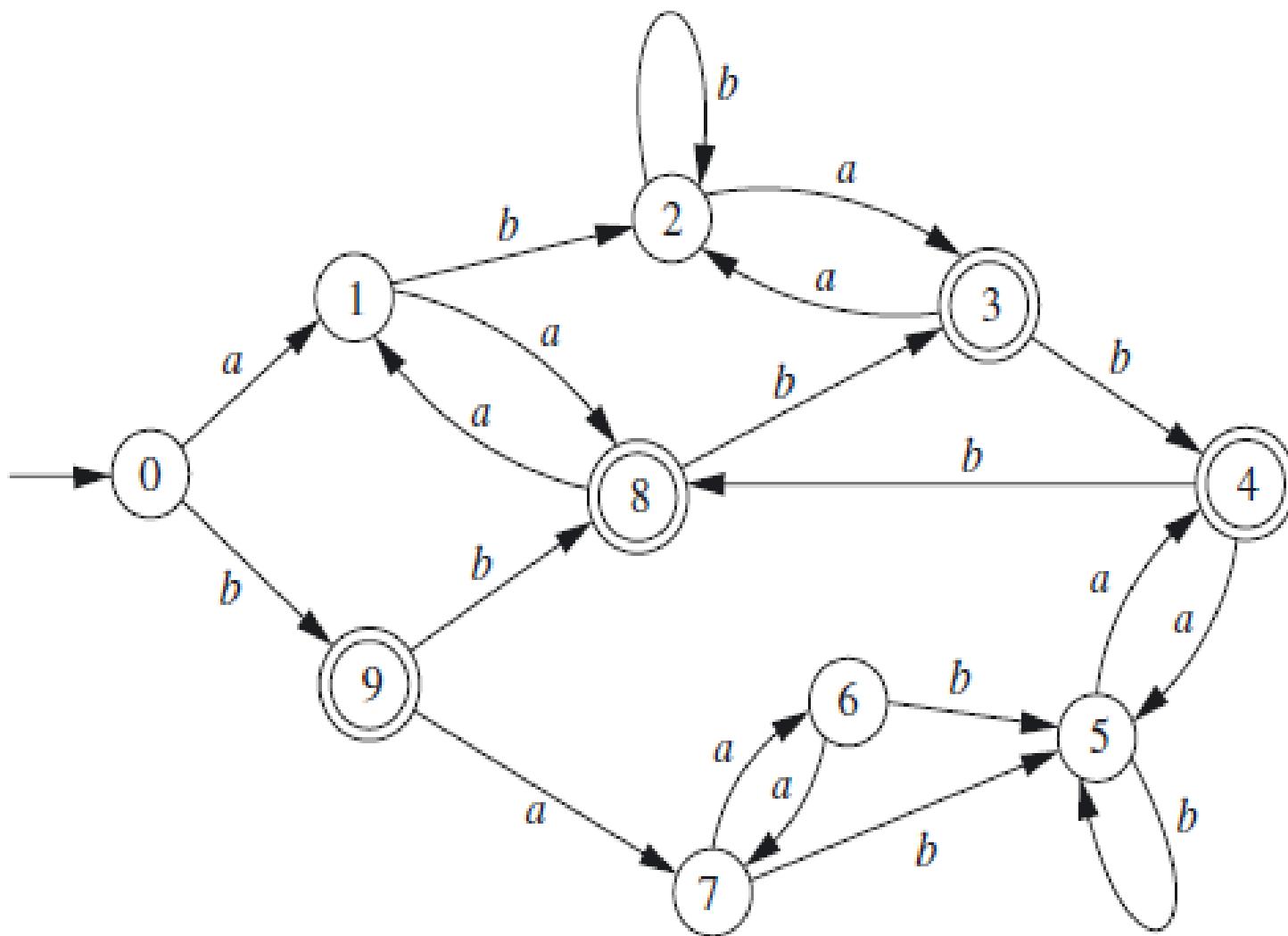
$$\delta(f, 0) = f \quad \delta(f, 1) = f$$



# Example

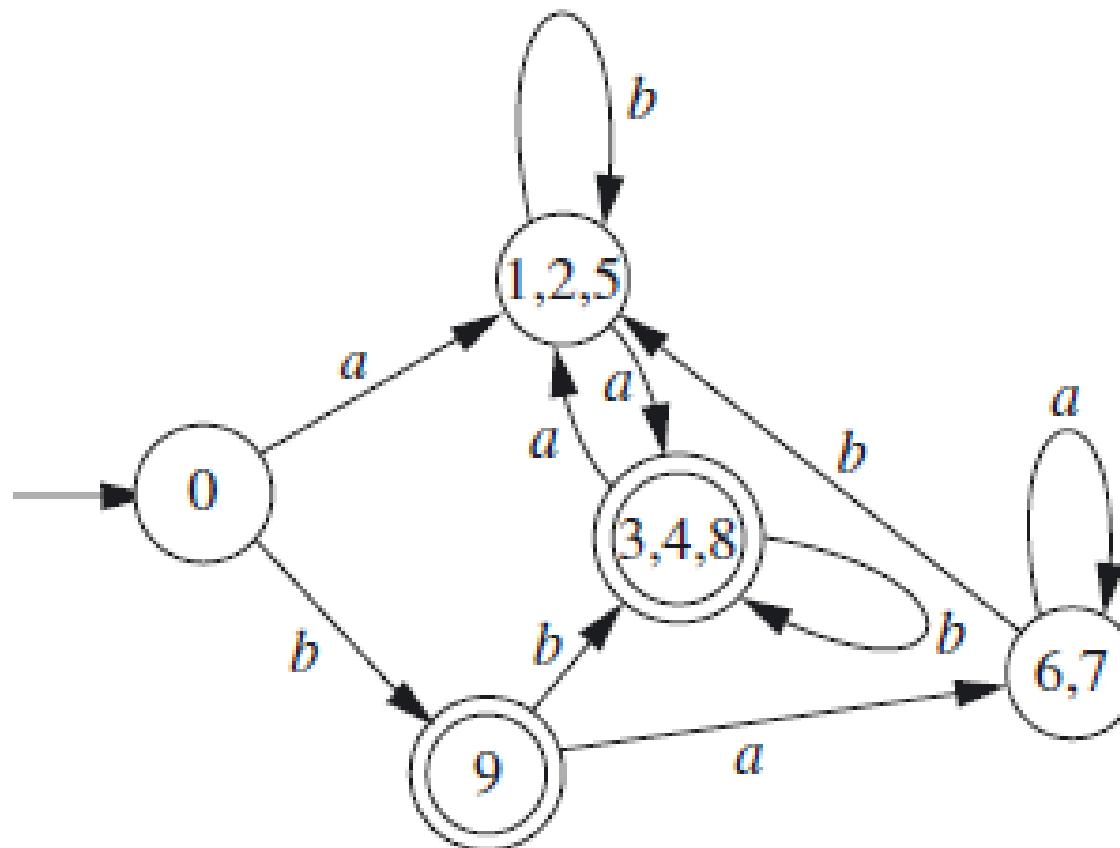


# Minimize following DFA



1	2							
2	2							
3	1	1	1					
4	1	1	1					
5	2			1	1			
6	2	2	2	1	1	2		
7	2	2	2	1	1	2		
8	1	1	1			1	1	1
9	1	1	1	2	3	1	1	2
	0	1	2	3	4	5	6	7

# Final DFA



# Absent No 28/6/19

- 2,7,16,18,21,25,26,27,29,34,37,50,53,57,59,65,70,

# Recall

- NFA
- NFA-null

# The Language of an NFA $M$

The language accepted by  $M$  is:

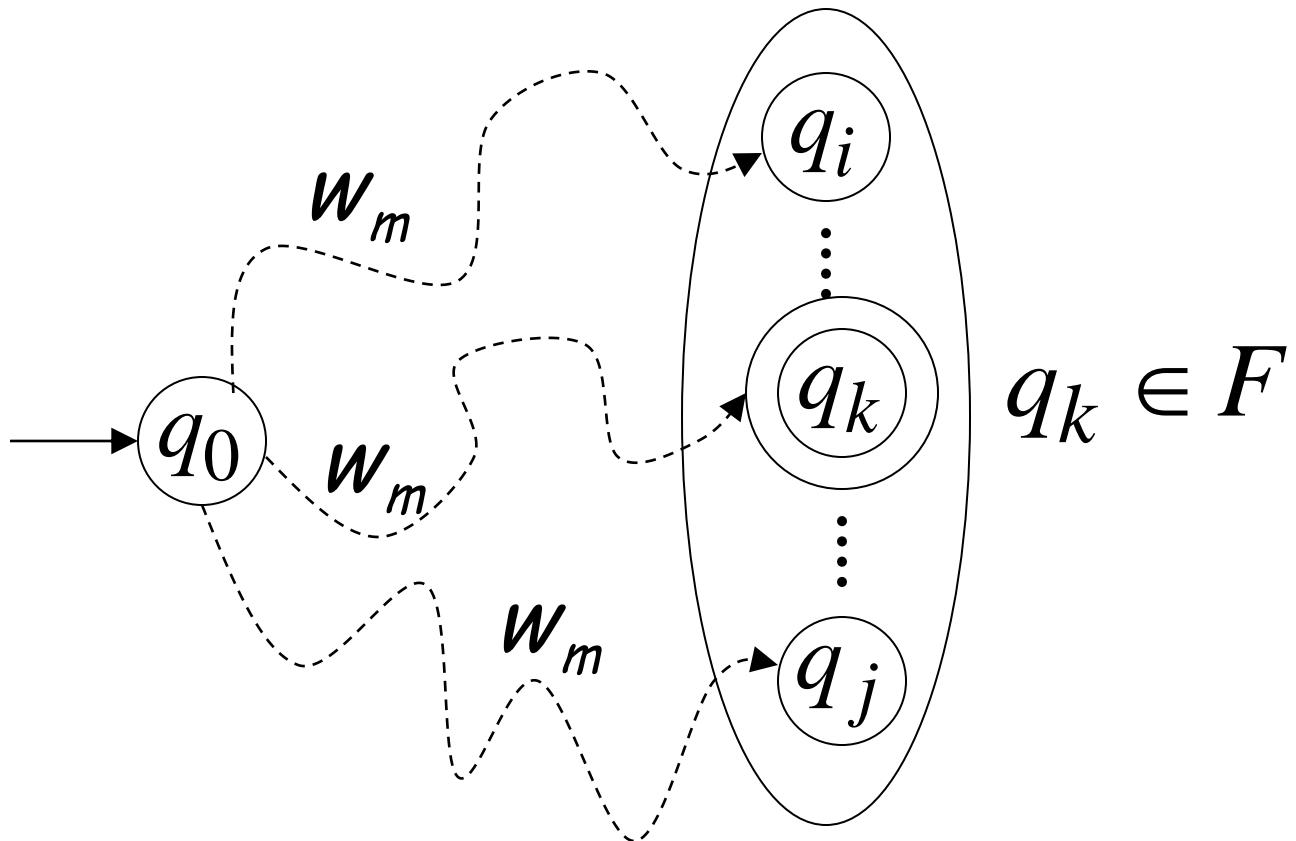
$$L(M) = \{w_1, w_2, \dots, w_n\}$$

where  $\delta^*(q_0, w_m) = \{q_i, \dots, q_k, \dots, q_j\}$

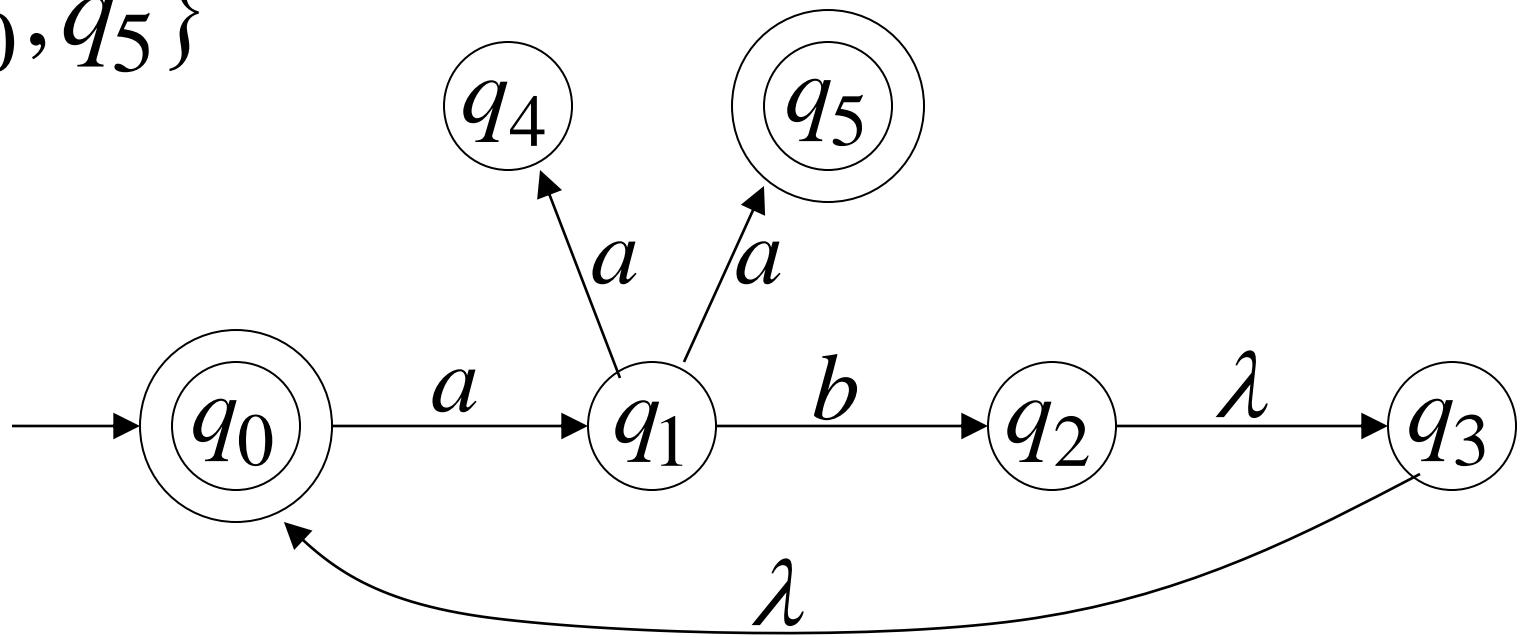
and there is some  $q_k \in F$  (accepting state)

$$w_m \in L(M)$$

$$\delta^*(q_0, w_m)$$



$$F = \{q_0, q_5\}$$

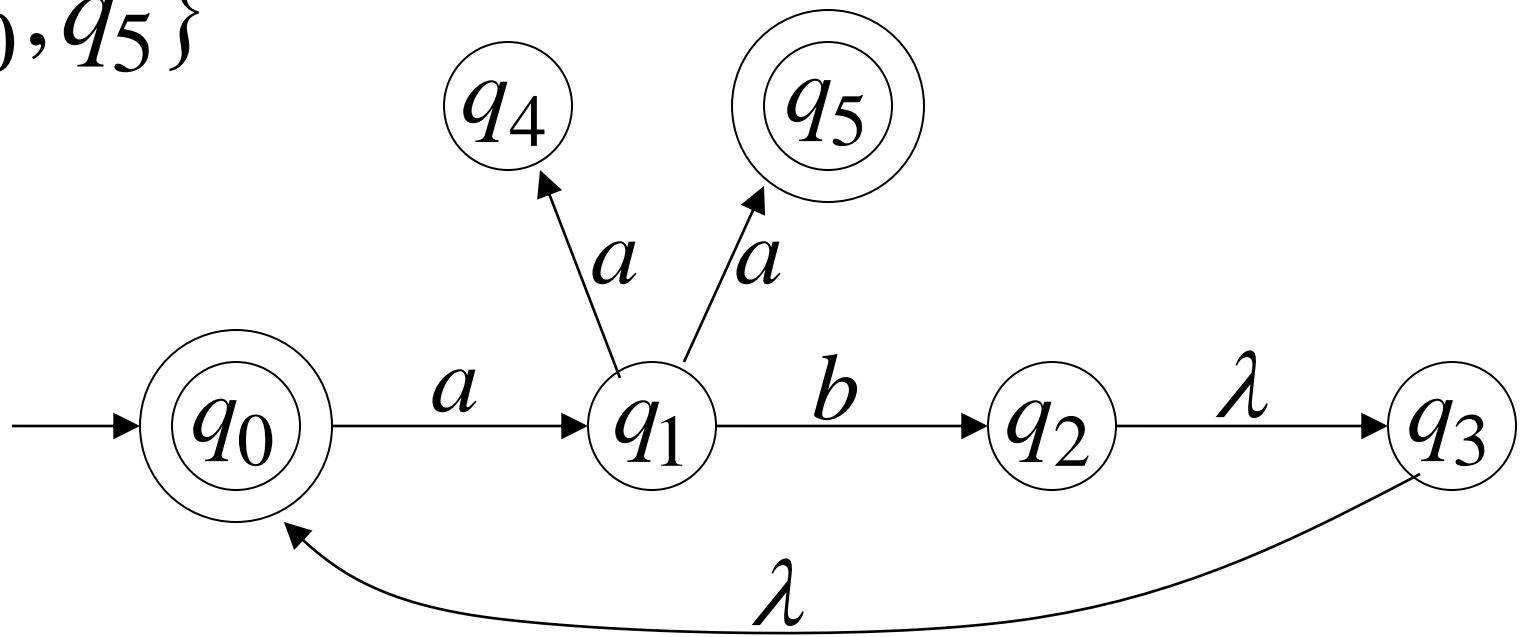


$$\delta^*(q_0, aa) = \{q_4, \underline{q_5}\} \in F$$

$\delta^*(q_0, aa) = \{q_4, \underline{q_5}\} \in F$

$aa \in L(M)$

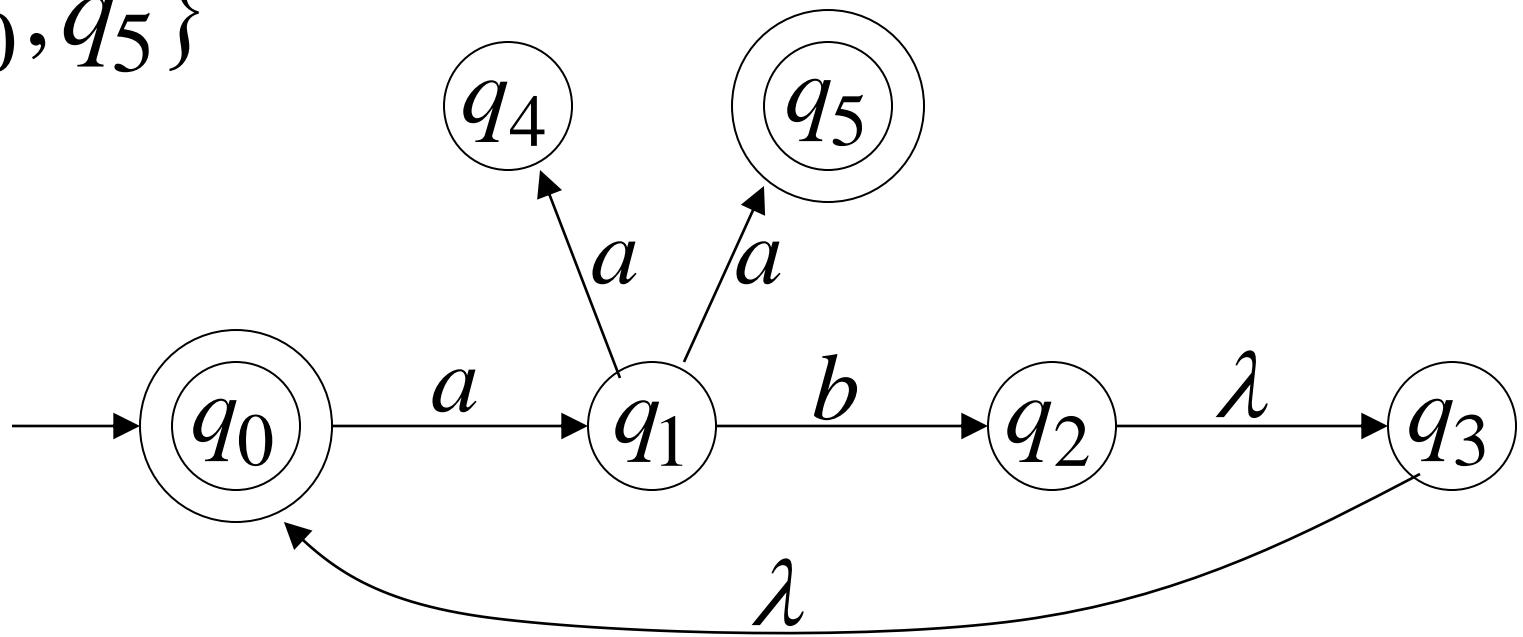
$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, ab) = \{q_2, q_3, \underline{q_0}\} \quad ab \in L(M)$$

$\in F$

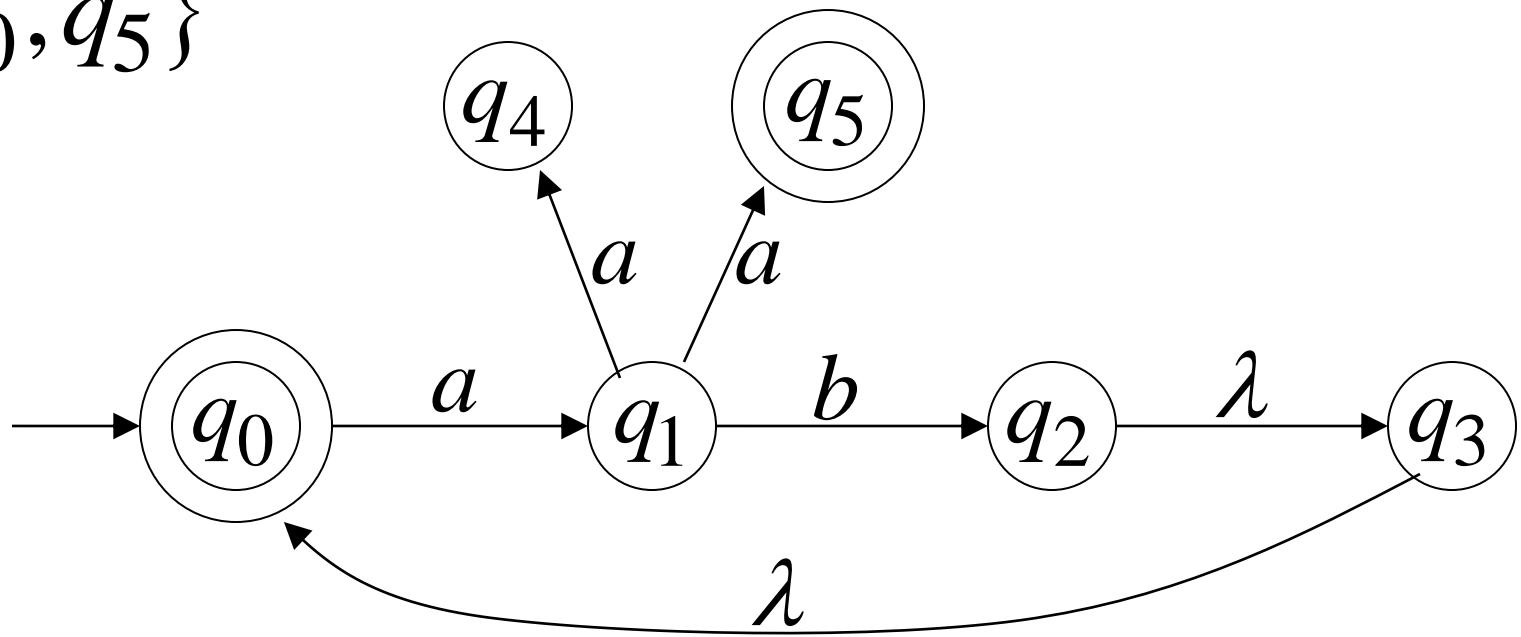
$$F = \{q_0, q_5\}$$



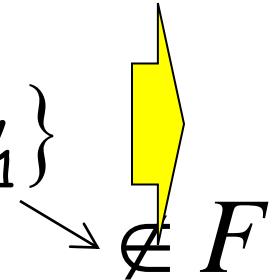
$$\delta^*(q_0, aba) = \{q_4, \underline{q_5}\} \xrightarrow{\quad \in F \quad}$$

$$aaba \in L(M)$$

$$F = \{q_0, q_5\}$$

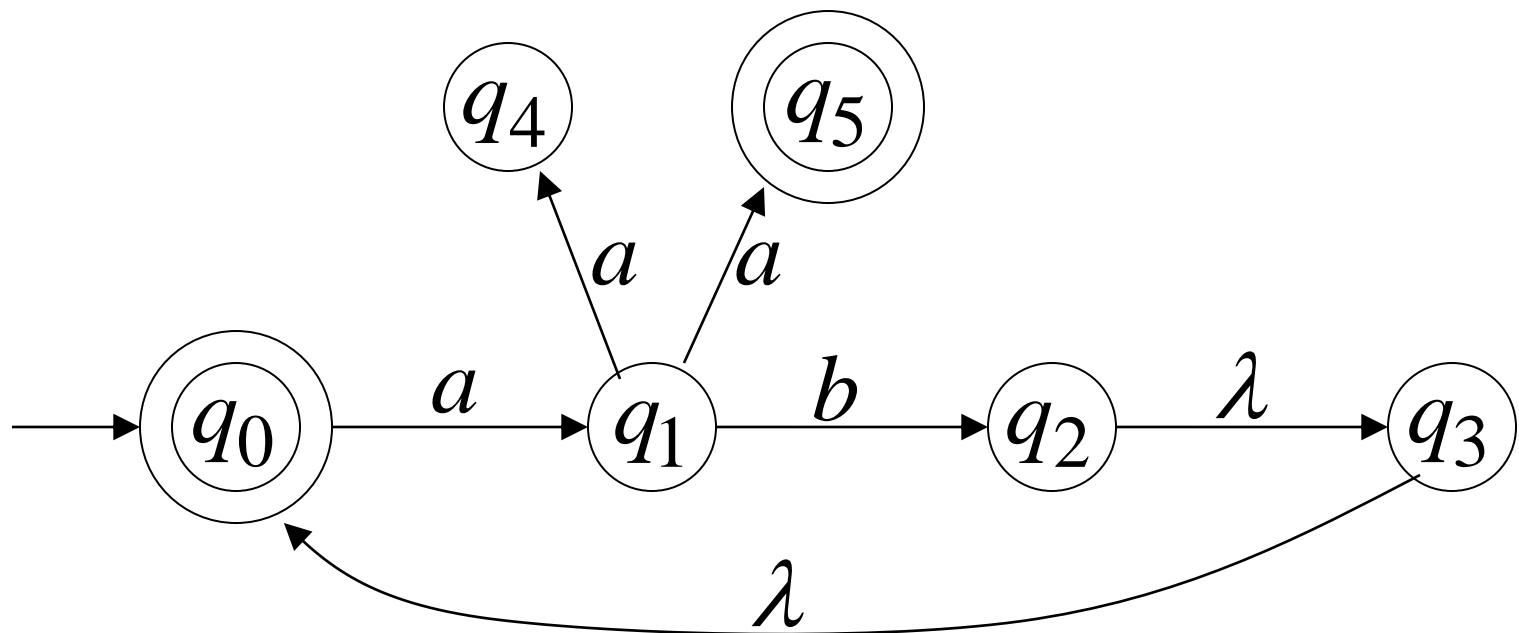


$$\delta^*(q_0, aba) = \{q_1\}$$



$$aba \notin L(M)$$

$$\notin F$$

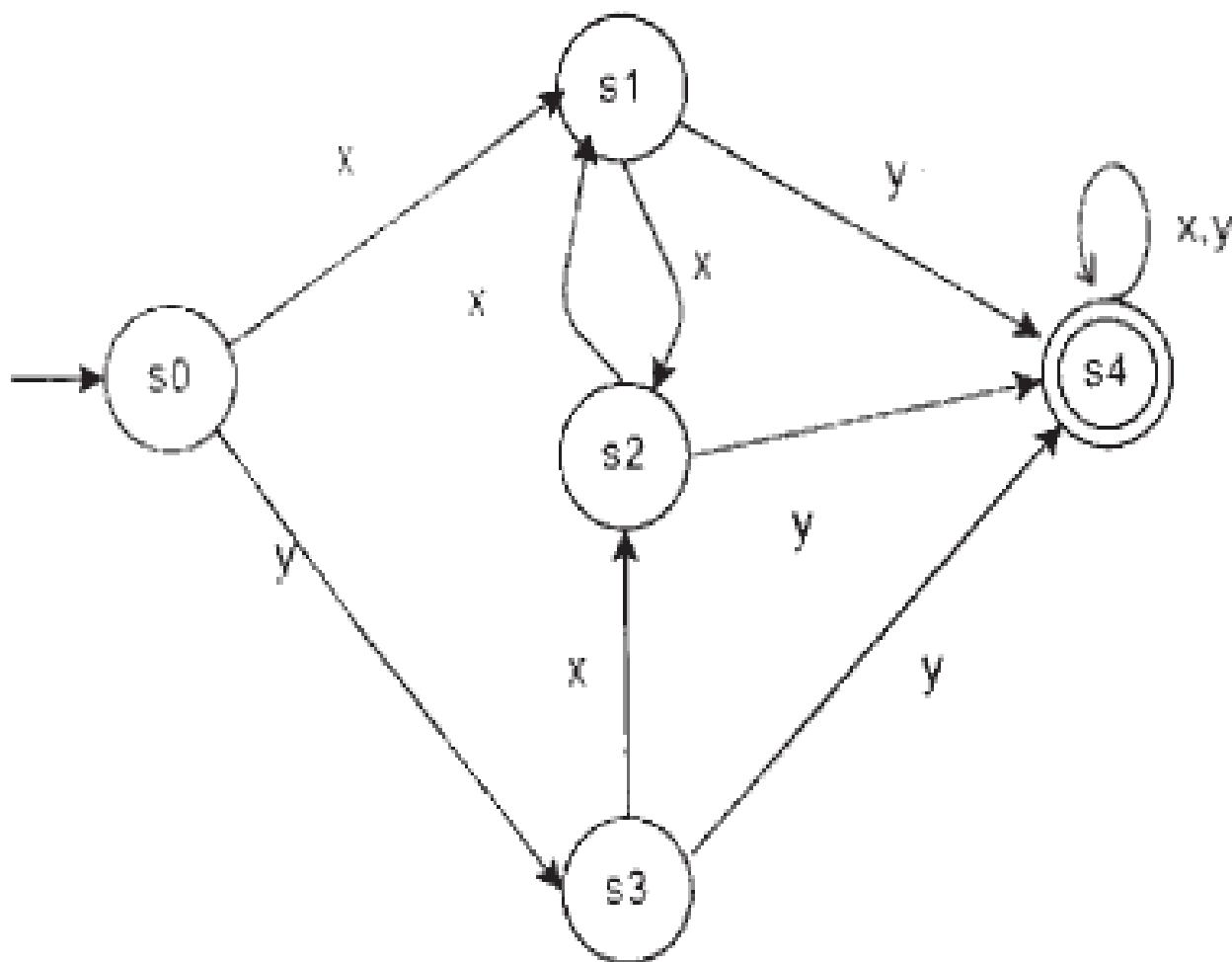


$$L(M) = \{ab\}^* \cup \{ab\}^* \{aa\}$$

# Recall

- NFA
- NFA-null
- Minimization of DFa

Minimize the following automata.



NFAs accept the Regular  
Languages

# Equivalence of Machines

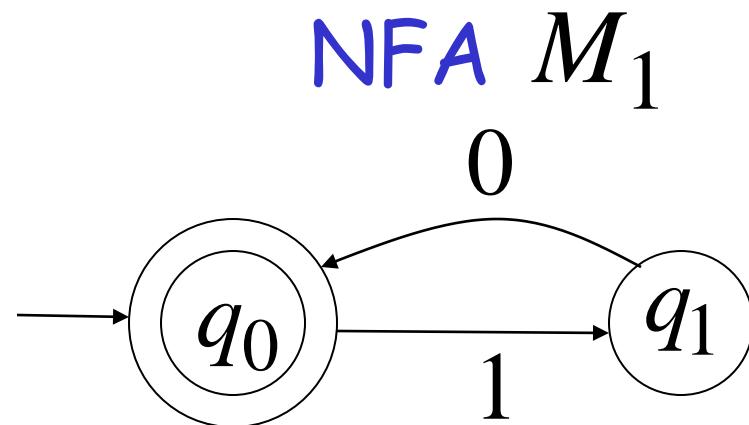
Definition:

Machine  $M_1$  is equivalent to machine  $M_2$

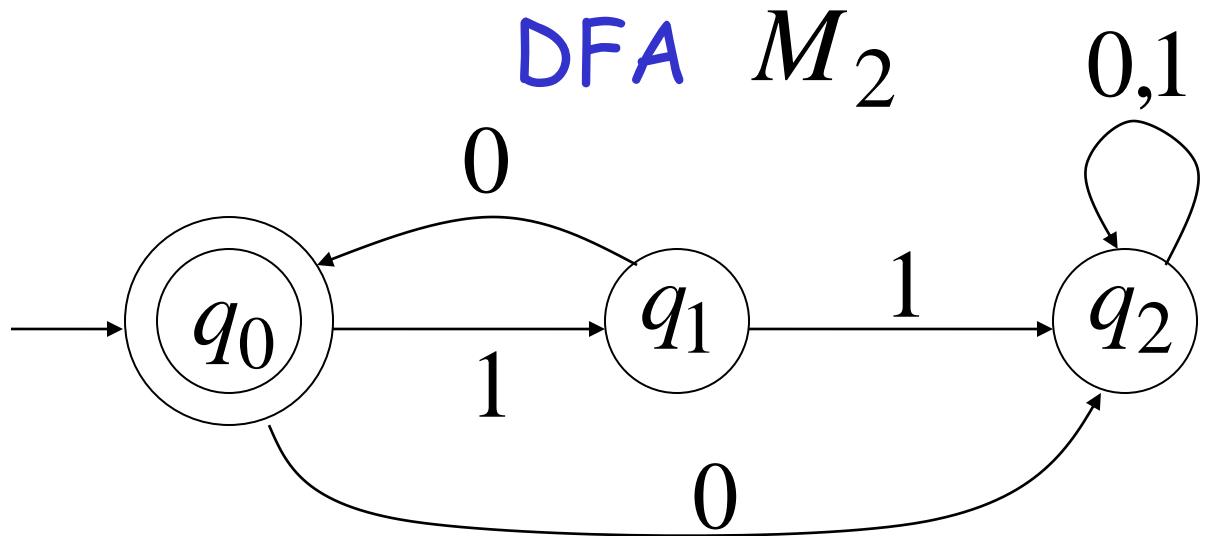
if  $L(M_1) = L(M_2)$

# Example of equivalent machines

$$L(M_1) = \{10\}^*$$



$$L(M_2) = \{10\}^*$$



# Theorem:

$$\left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by NFAs} \end{array} \right\} = \left\{ \begin{array}{l} \text{Regular} \\ \text{Languages} \end{array} \right\}$$

Languages  
accepted  
by DFAs

NFAs and DFAs have the same computation power,  
accept the same set of languages

**Proof:** we only need to show

$$\left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by NFAs} \end{array} \right\} \supseteq \left\{ \begin{array}{l} \text{Regular} \\ \text{Languages} \end{array} \right\}$$

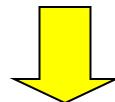
AND

$$\left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by NFAs} \end{array} \right\} \subseteq \left\{ \begin{array}{l} \text{Regular} \\ \text{Languages} \end{array} \right\}$$

## Proof-Step 1

$$\left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by NFAs} \end{array} \right\} \equiv \left\{ \begin{array}{l} \text{Regular} \\ \text{Languages} \end{array} \right\}$$

Every DFA is trivially an NFA

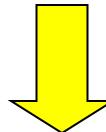


Any language  $L$  accepted by a DFA  
is also accepted by an NFA

## Proof-Step 2

$$\left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by NFAs} \end{array} \right\} \subseteq \left\{ \begin{array}{l} \text{Regular} \\ \text{Languages} \end{array} \right\}$$

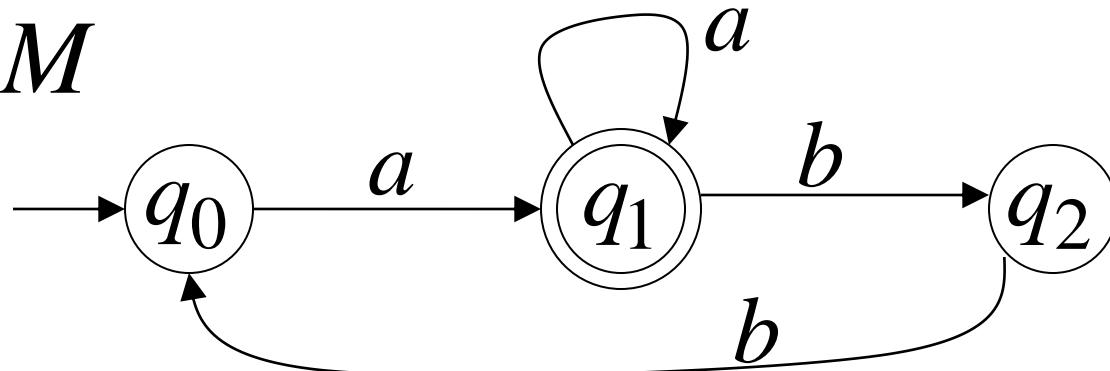
Any NFA can be converted to an equivalent DFA



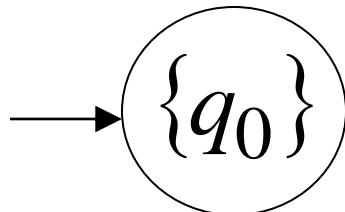
Any language  $L$  accepted by an NFA is also accepted by a DFA

# Conversion NFA to DFA

NFA  $M$

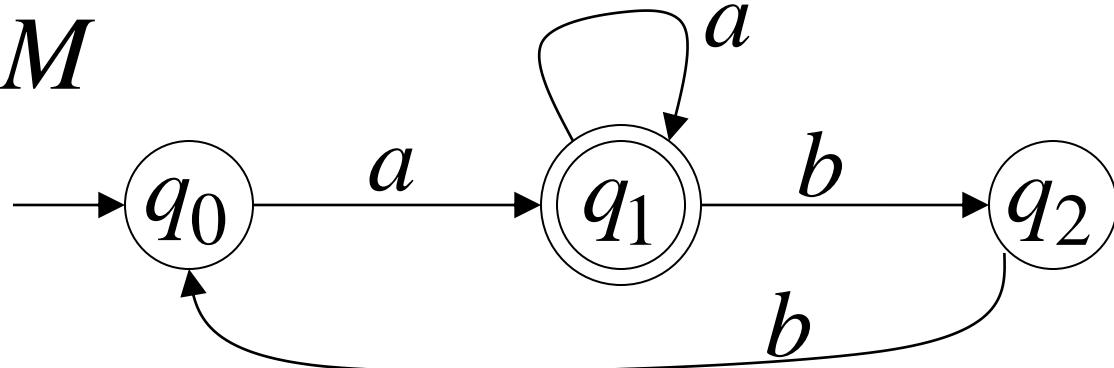


DFA  $M'$

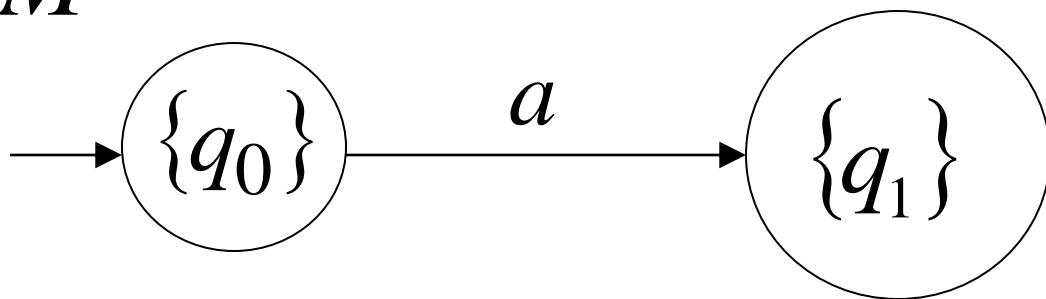


$$\delta^*(q_0, a) = \{q_1\}$$

NFA  $M$

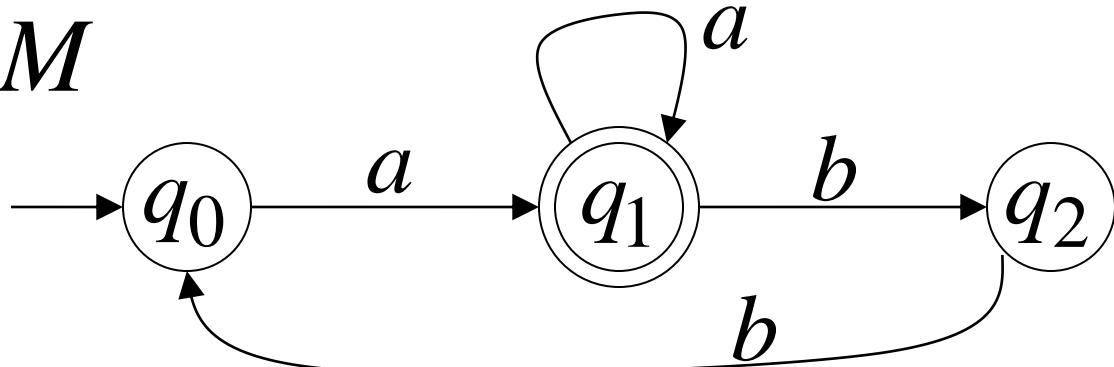


DFA  $M'$

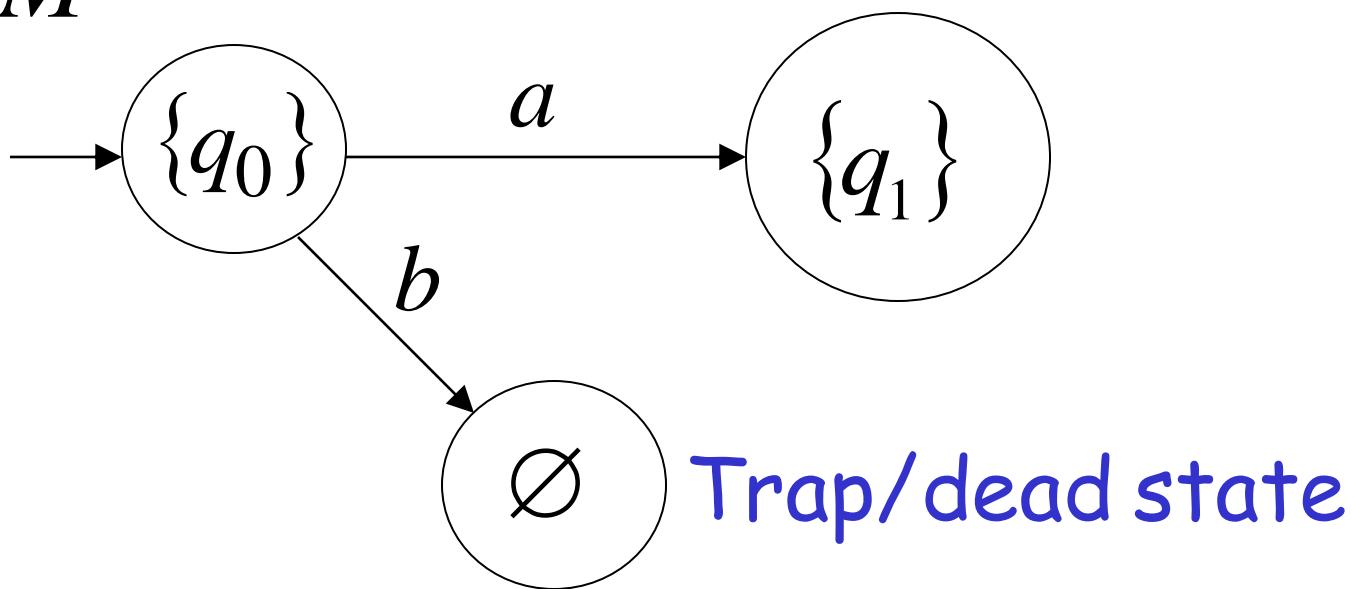


$$\delta^*(q_0, b) = \emptyset \quad \text{empty set}$$

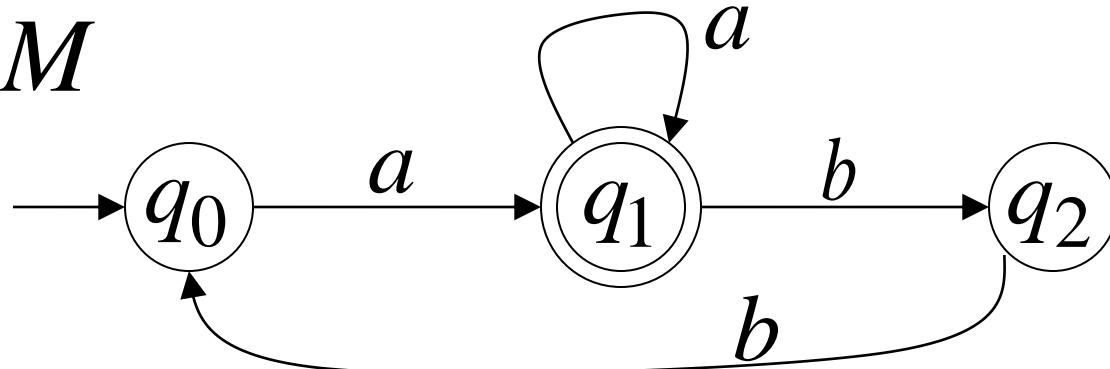
NFA  $M$



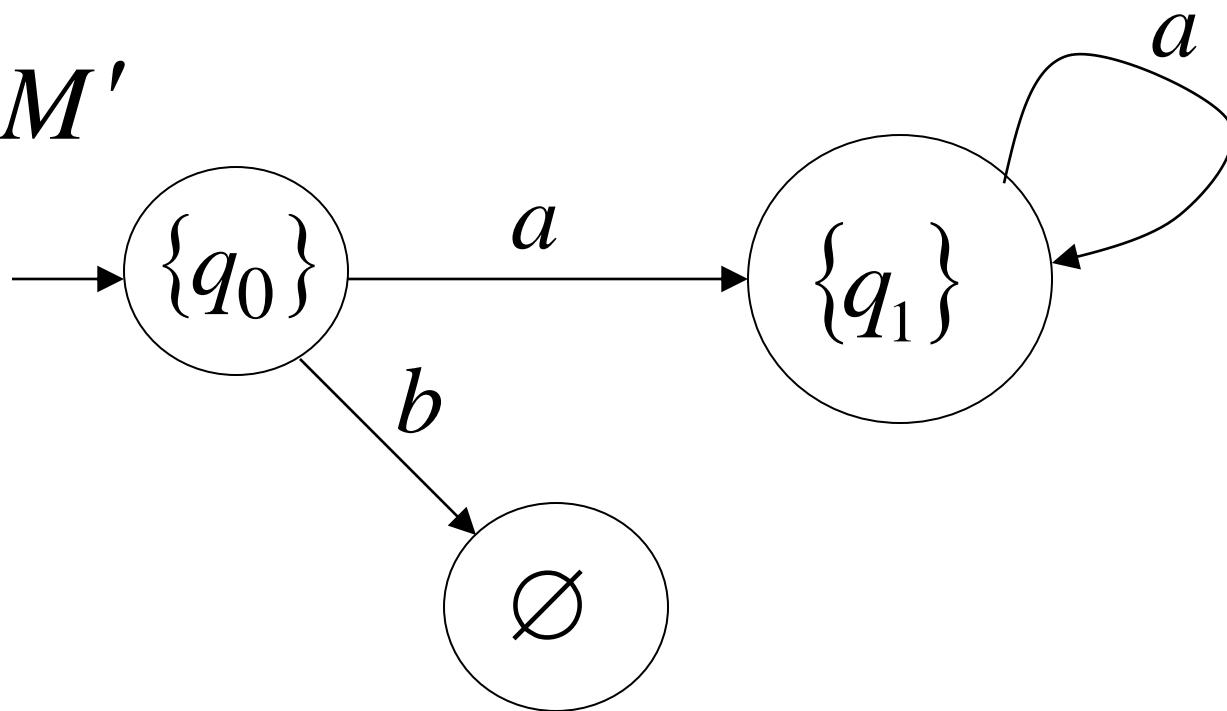
DFA  $M'$



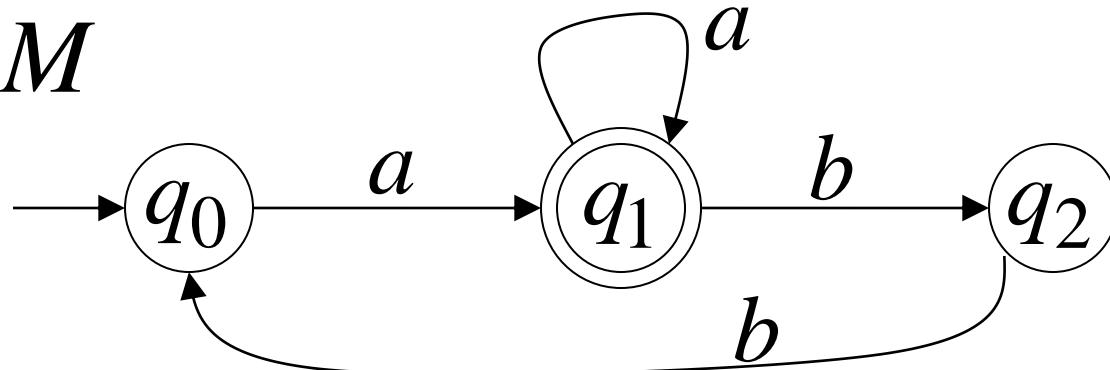
**NFA**  $M$



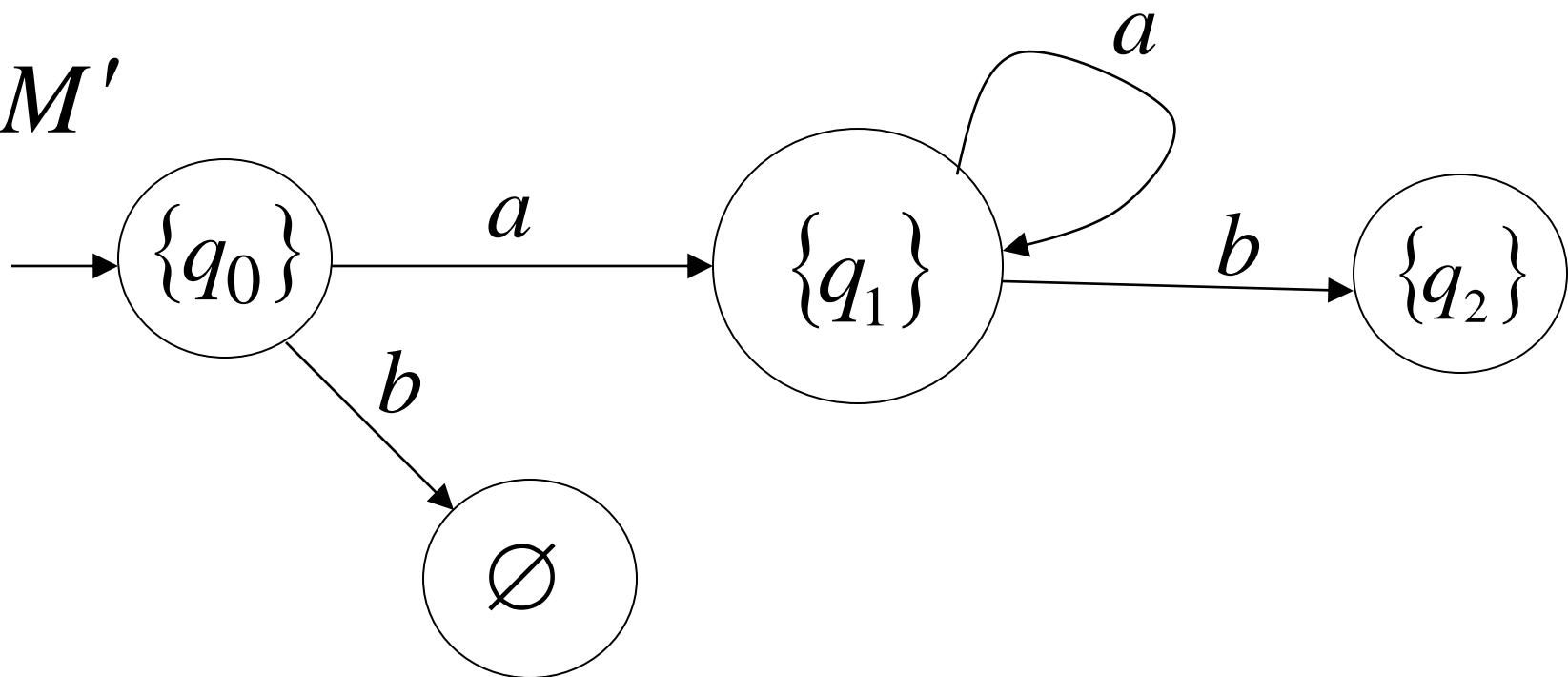
**DFA**  $M'$



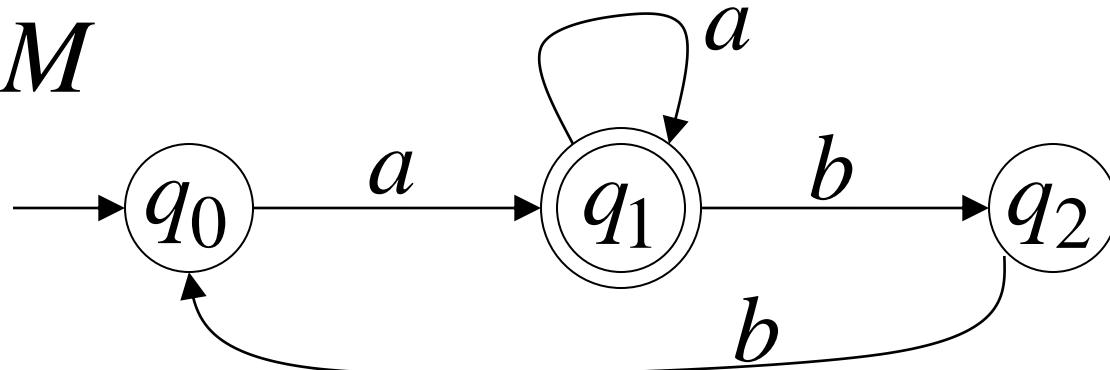
**NFA**  $M$



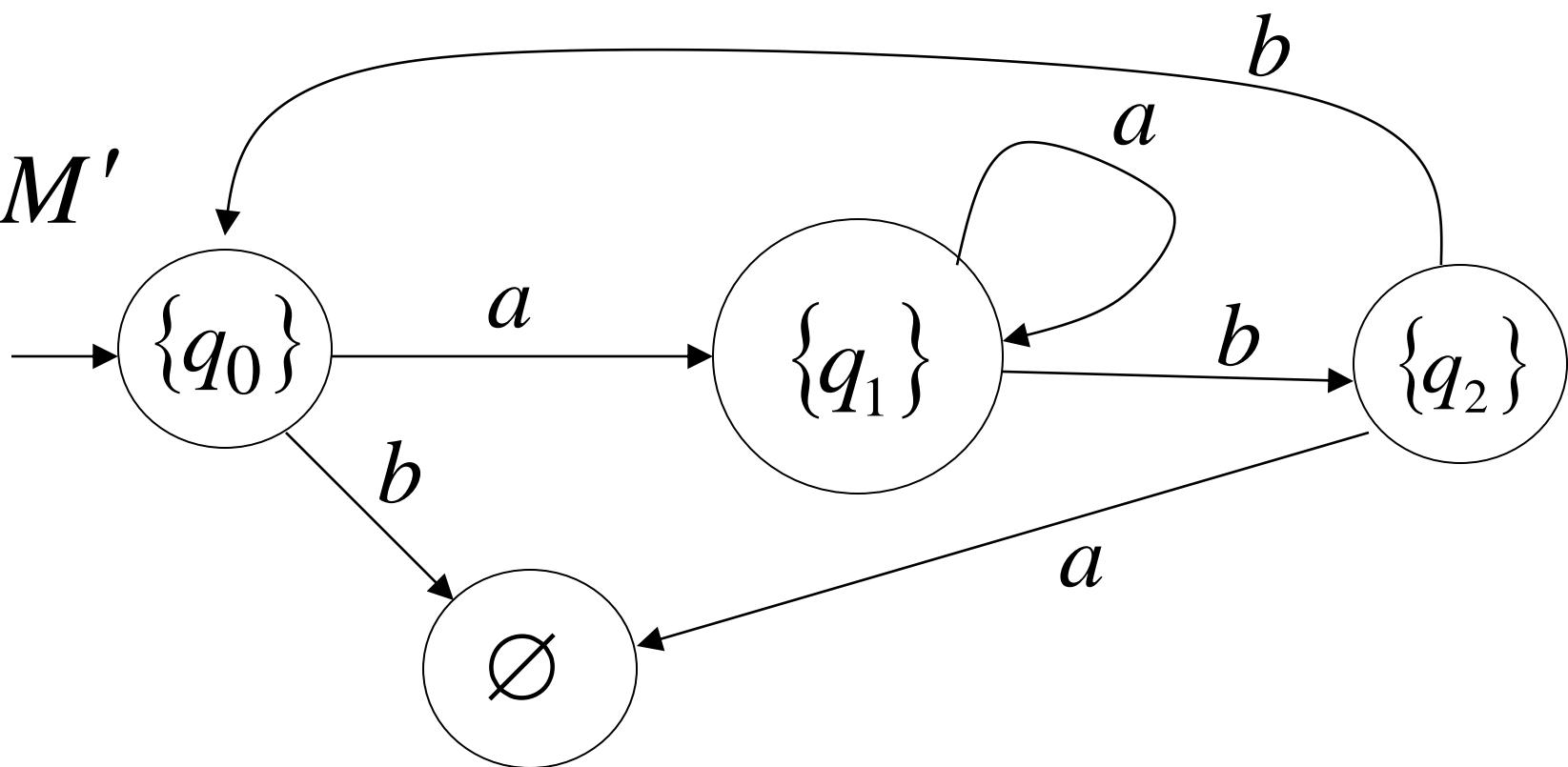
**DFA**  $M'$



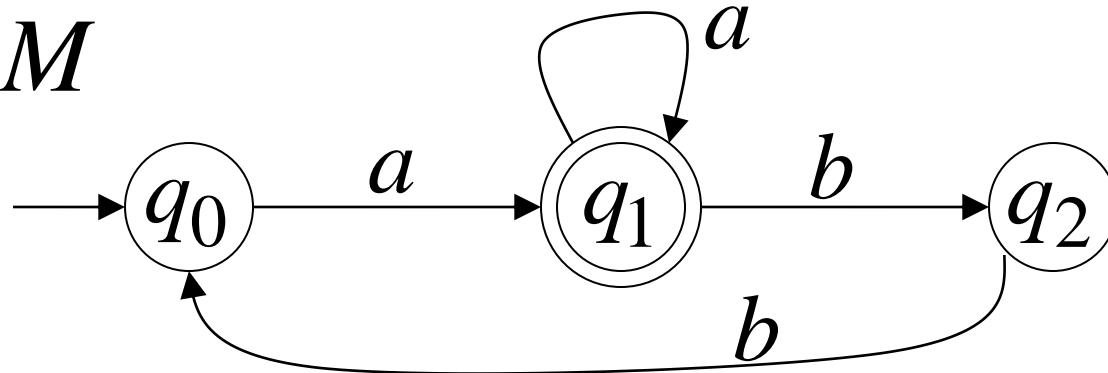
NFA  $M$



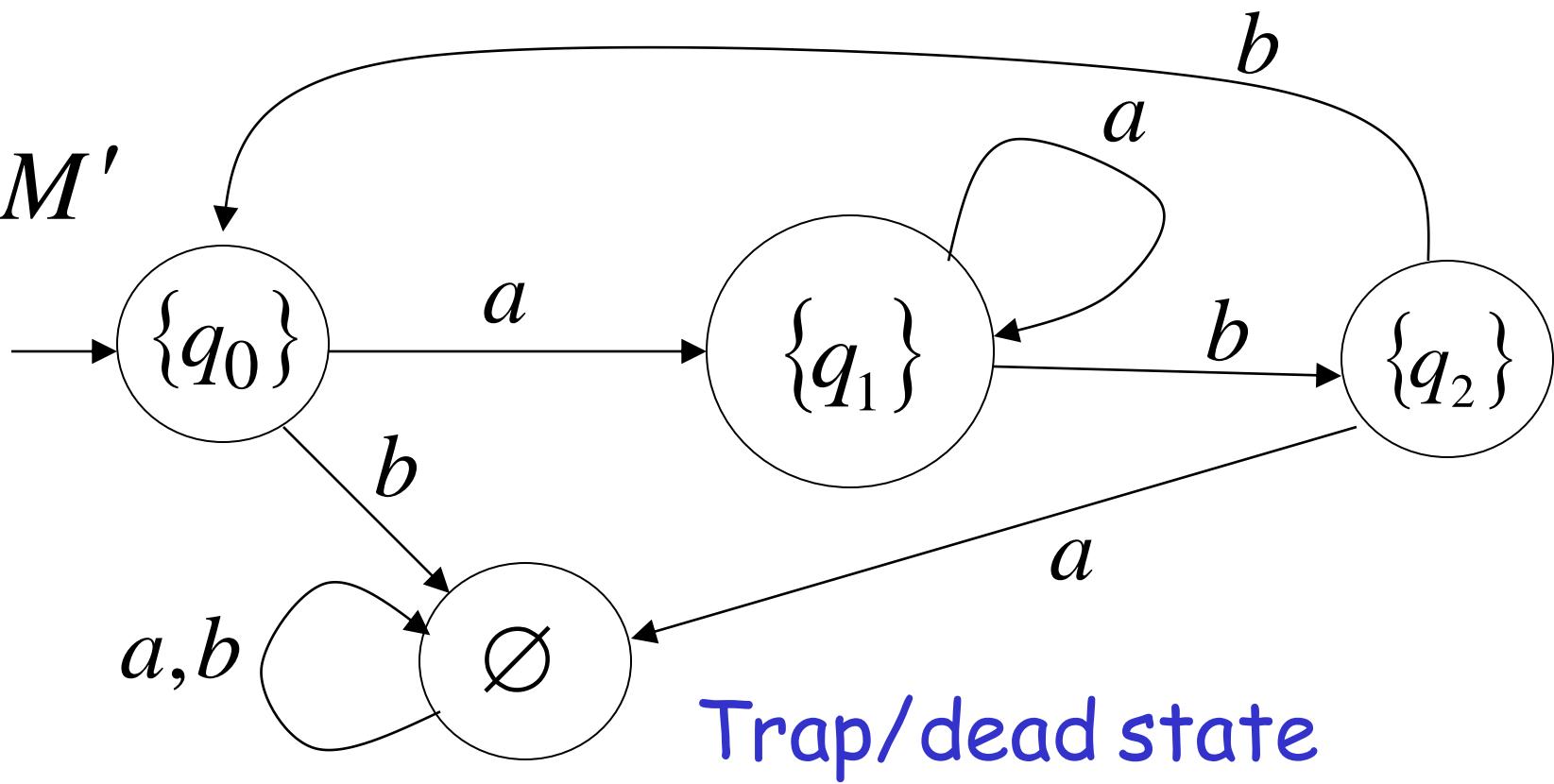
DFA  $M'$



NFA  $M$

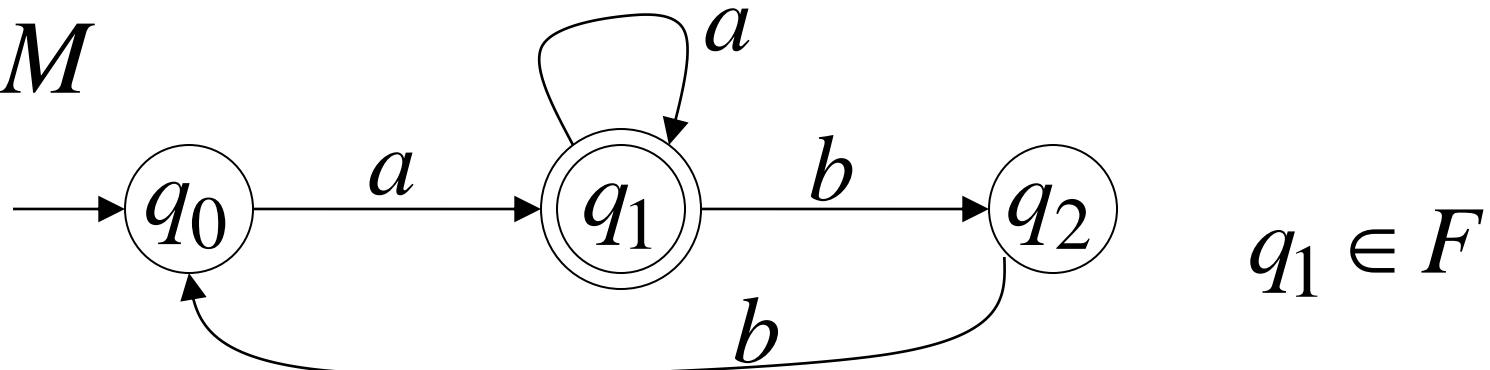


DFA  $M'$



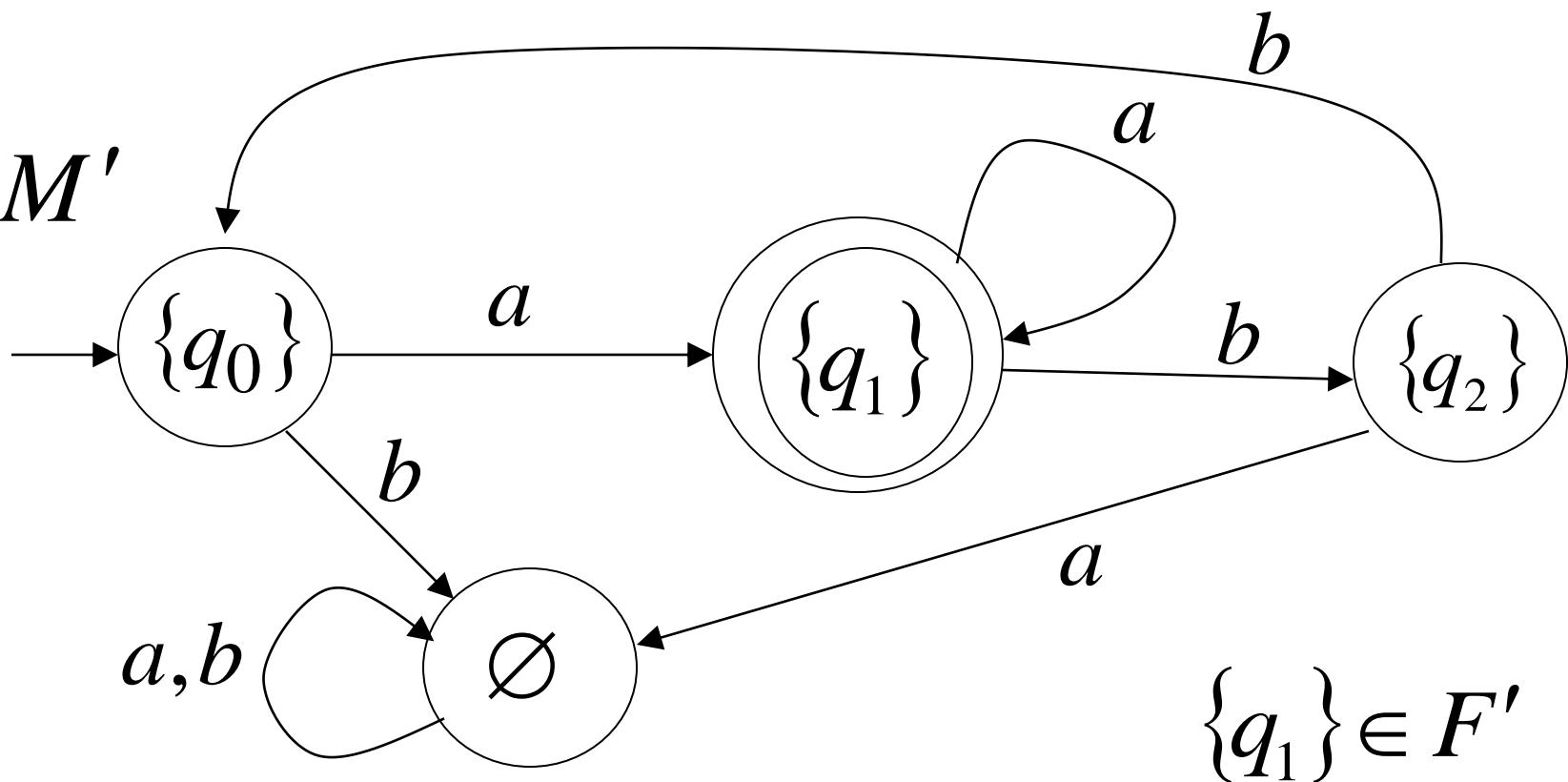
# END OF CONSTRUCTION

NFA  $M$



$$q_1 \in F$$

DFA  $M'$



$$\{q_1\} \in F'$$

# Conversion NFA to DFA

# General Conversion Procedure

Input: an NFA  $M$

Output: an equivalent DFA  $M'$   
with  $L(M) = L(M')$

The NFA has states  $q_0, q_1, q_2, \dots$

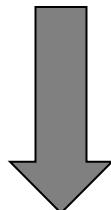
The DFA has states from the power set

$\emptyset, \{q_0\}, \{q_1\}, \{q_0, q_1\}, \{q_1, q_2, q_3\}, \dots$

# Conversion Procedure Steps

step

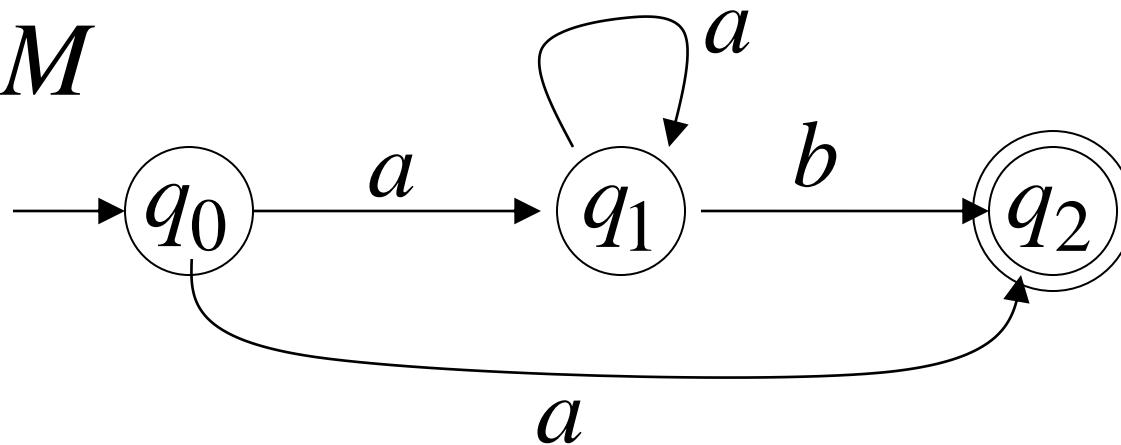
1. Initial state of NFA:  $q_0$



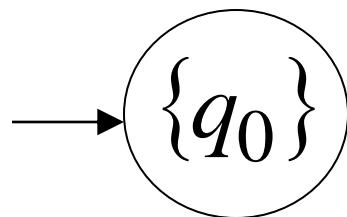
Initial state of DFA:  $\{q_0\}$

# Example

NFA  $M$

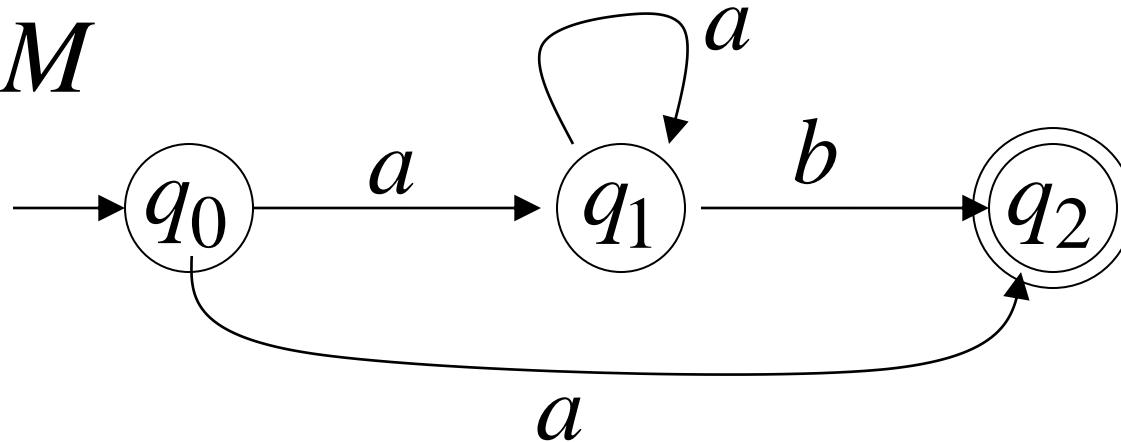


DFA  $M'$

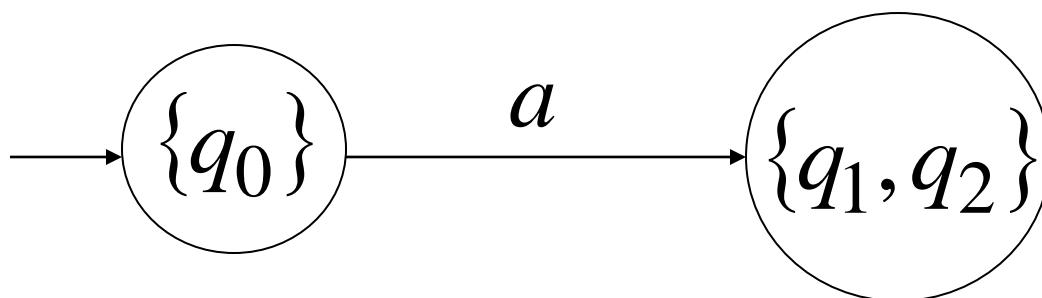


# Example

NFA  $M$

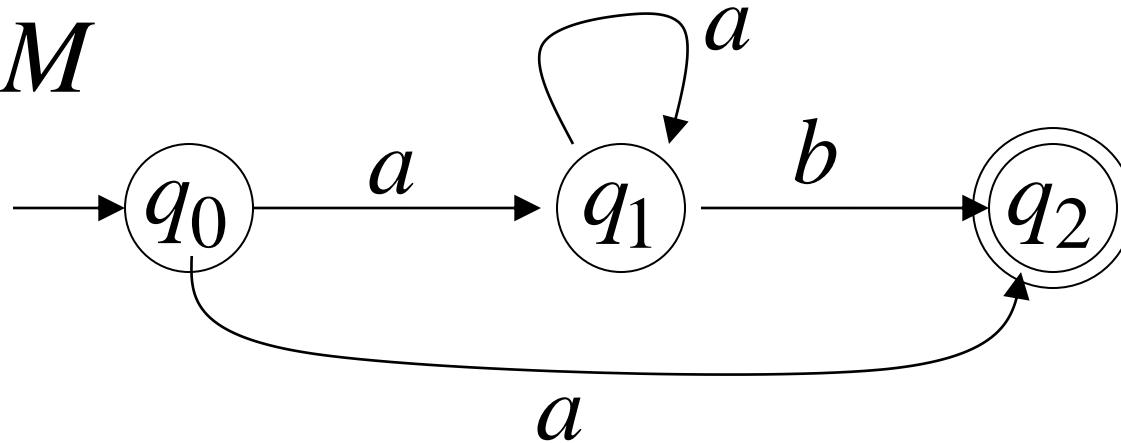


DFA  $M'$      $\delta(\{q_0\}, a) = \{q_1, q_2\}$

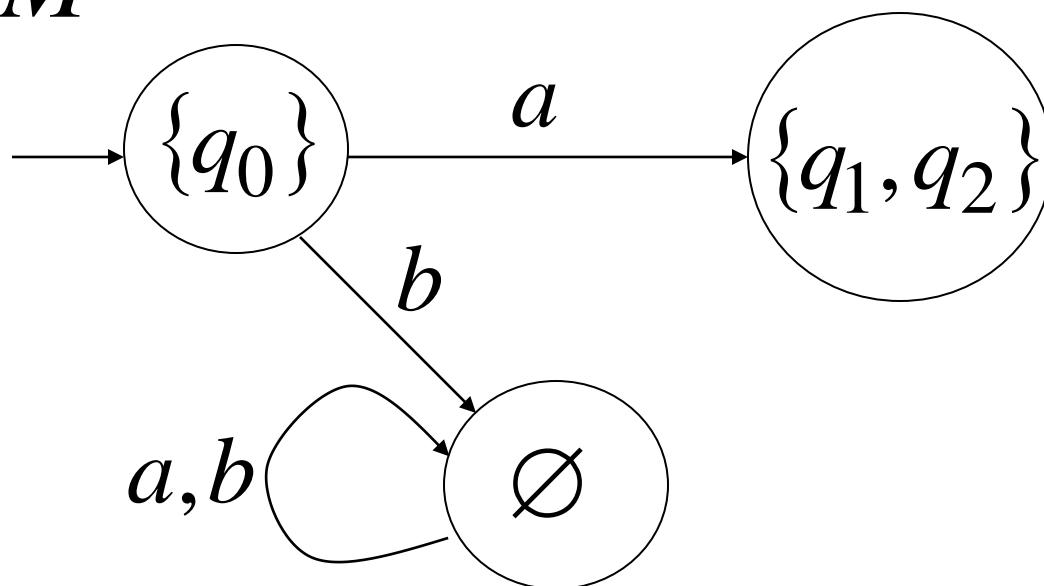


# Example

NFA  $M$



DFA  $M'$



step

2. For every DFA's state  $\{q_i, q_j, \dots, q_m\}$

compute in the NFA

$$\left. \begin{array}{l} \delta^*(q_i, a) \\ \cup \delta^*(q_j, a) \\ \dots \\ \cup \delta^*(q_m, a) \end{array} \right\} = \{q'_k, q'_l, \dots, q'_n\}$$

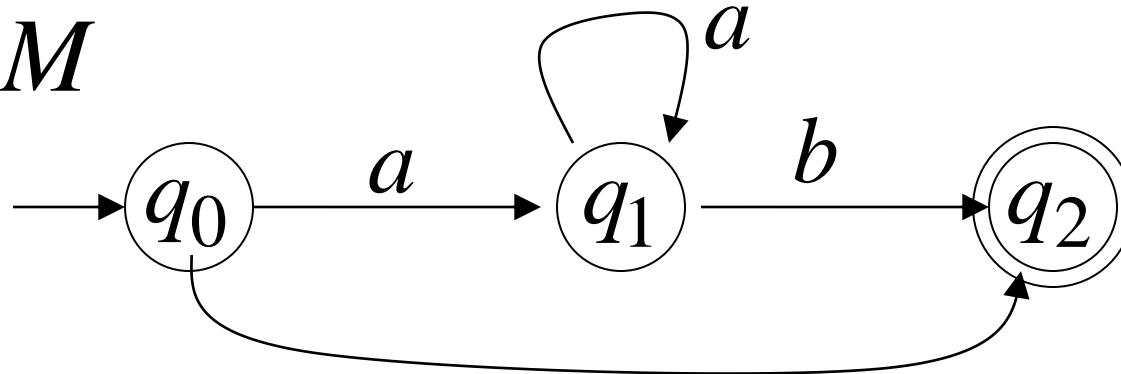
Union

add transition to DFA

$$\delta(\{q_i, q_j, \dots, q_m\}, a) = \{q'_k, q'_l, \dots, q'_n\}$$

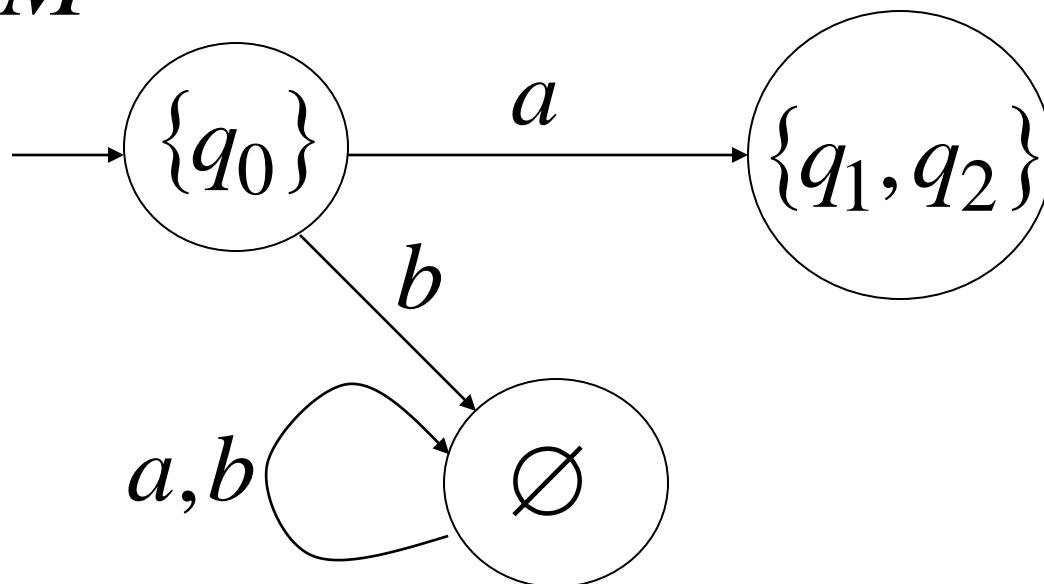
# Example

NFA  $M$



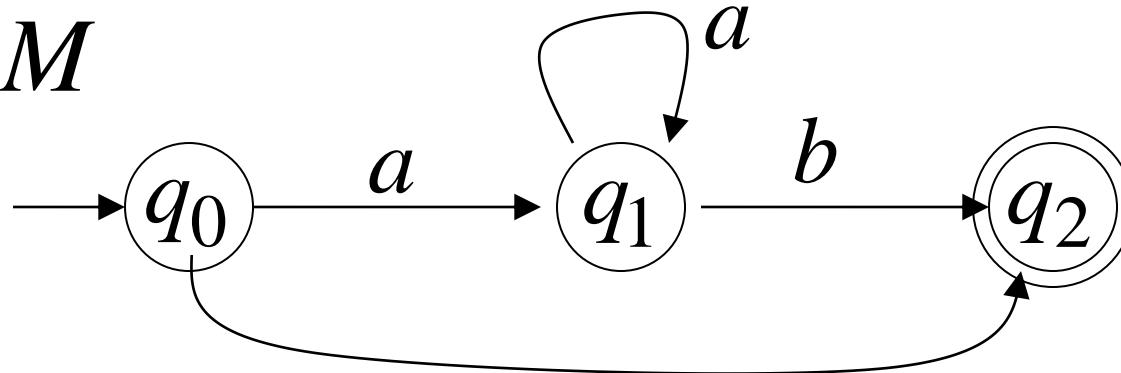
$$\delta^*(\{q_1, q_2\}, a) = \delta^*(q_1, a) \cup \delta^*(q_2, a) = \{q_1\} \cup \Theta = \{q_1\}$$

DFA  $M'$



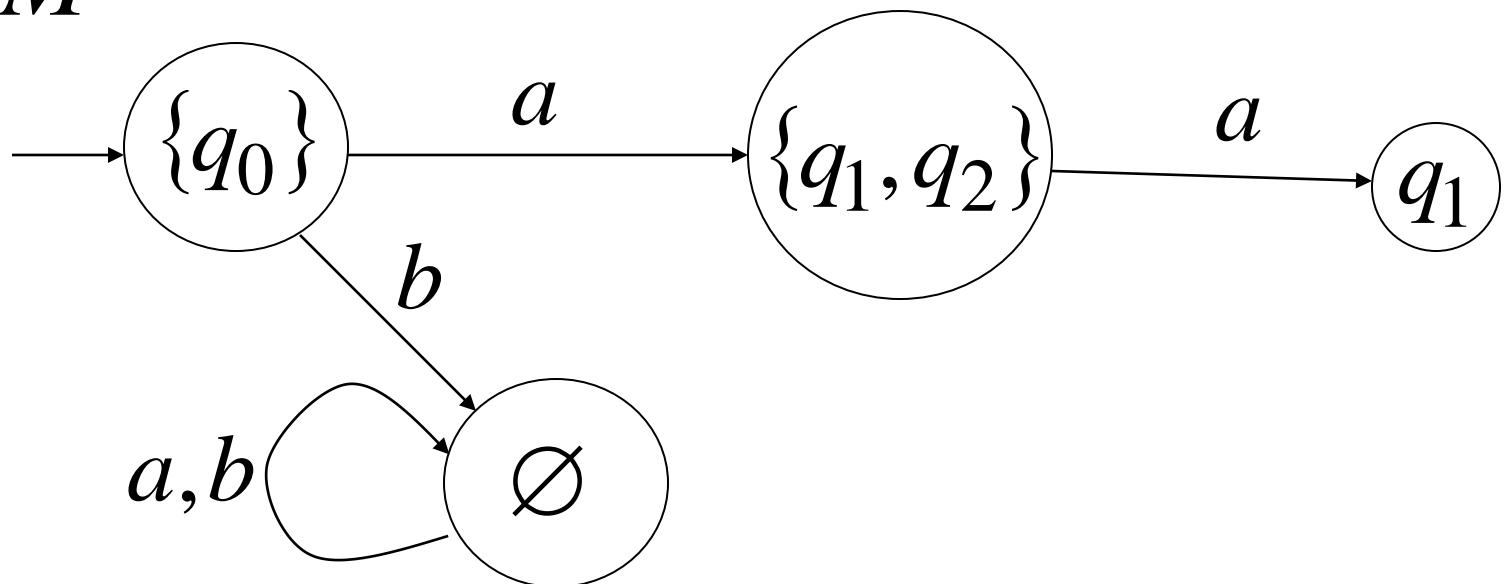
# Example

NFA  $M$



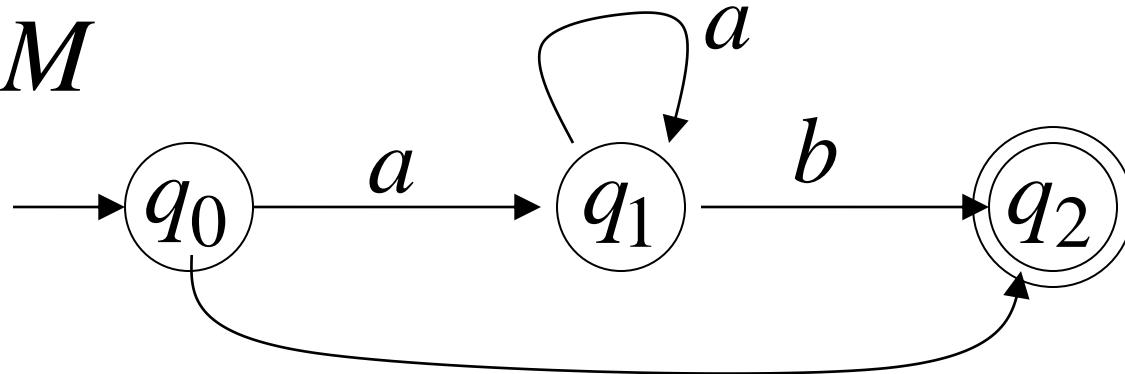
$$\delta^*(\{q_1, q_2\}, a) = \delta^*(q_1, a) \cup \delta^*(q_2, a) = \{q_1\} \cup \Theta = \{q_1\}$$

DFA  $M'$



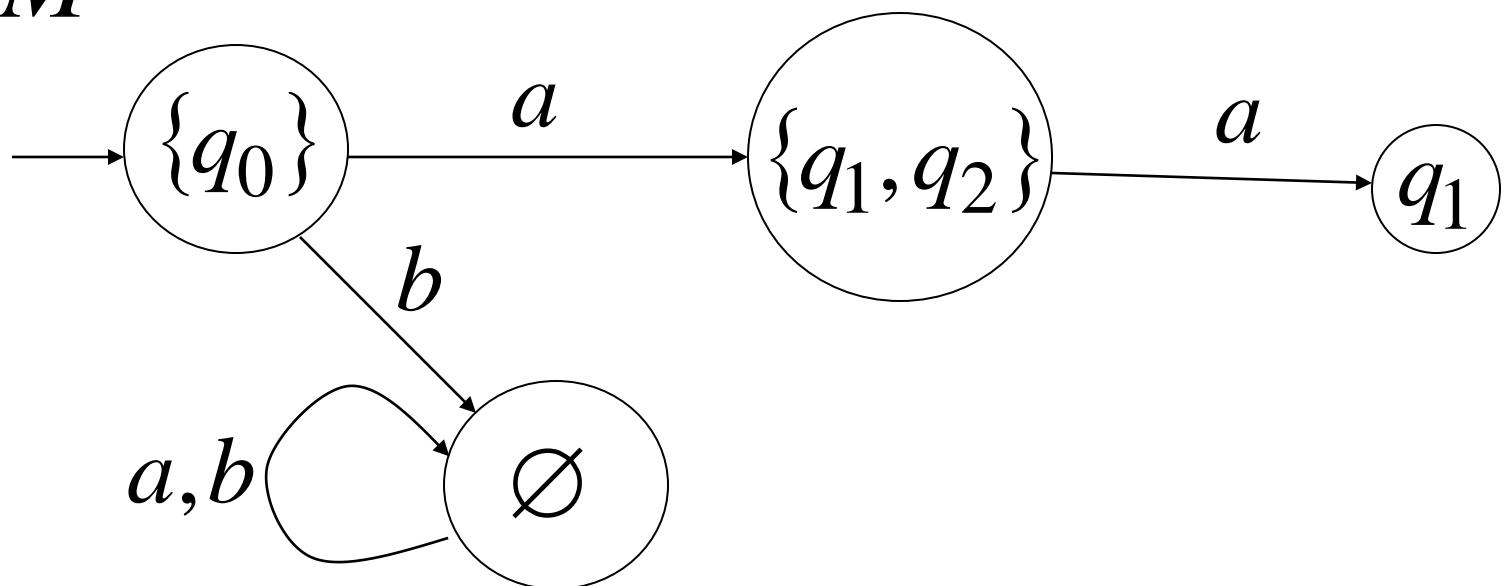
# Example

NFA  $M$



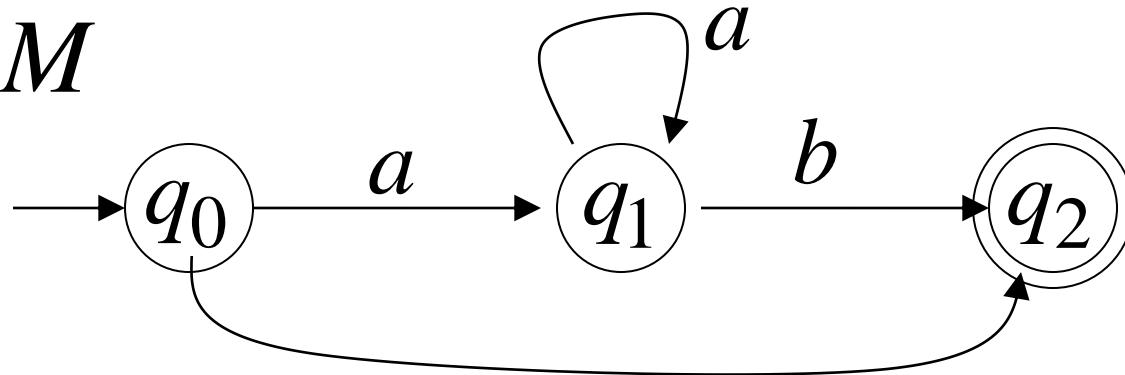
$$\delta^*(\{q_1, q_2\}, b) = \delta^*(q_1, b) \cup \delta^*(q_2, b) = \{q_2\} \cup \Theta = \{q_2\}$$

DFA  $M'$



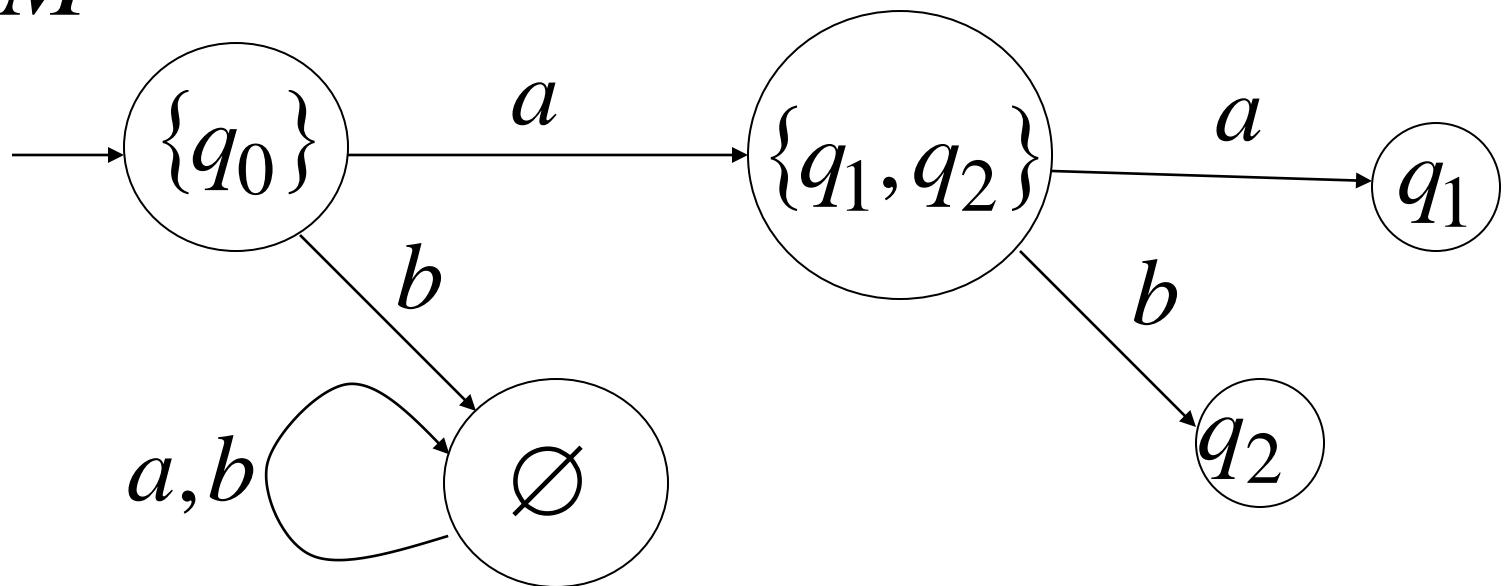
# Example

NFA  $M$



$$\delta^*(\{q_1, q_2\}, b) = \delta^*(q_1, b) \cup \delta^*(q_2, b) = \{q_2\} \cup \Theta = \{q_2\}$$

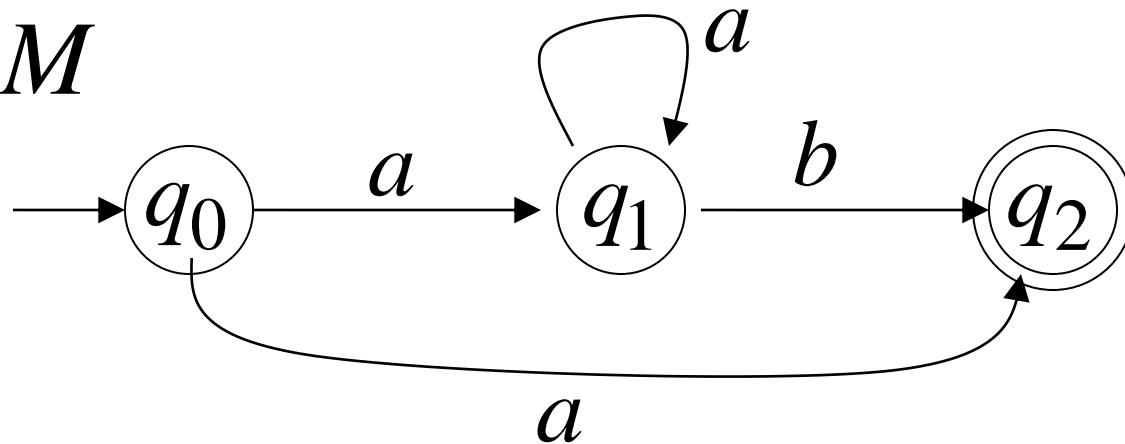
DFA  $M'$



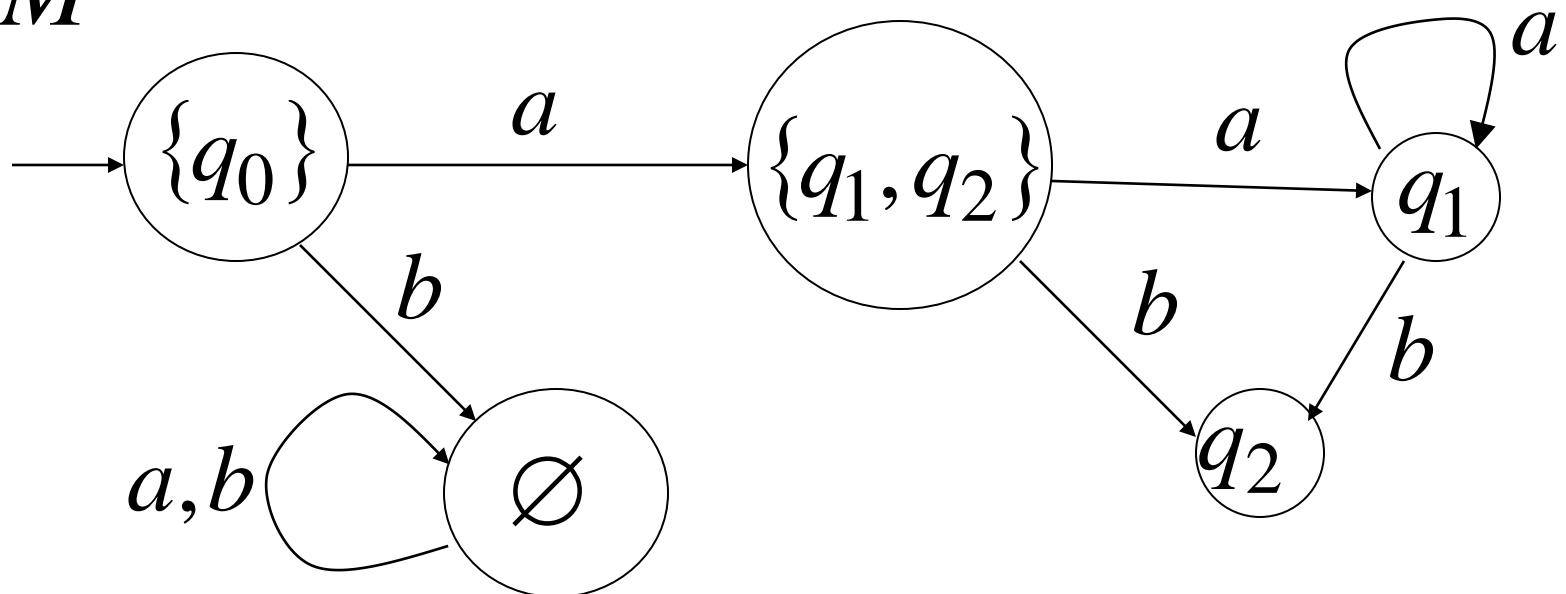
**step 3.** Repeat Step 2 for every state in DFA  
and symbols in alphabet until no more  
states can be added in the DFA

# Example

NFA  $M$

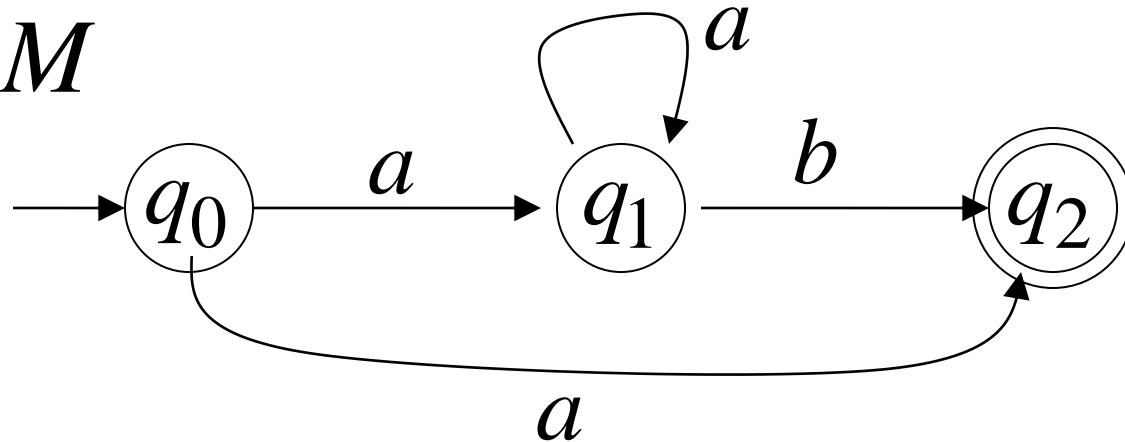


DFA  $M'$

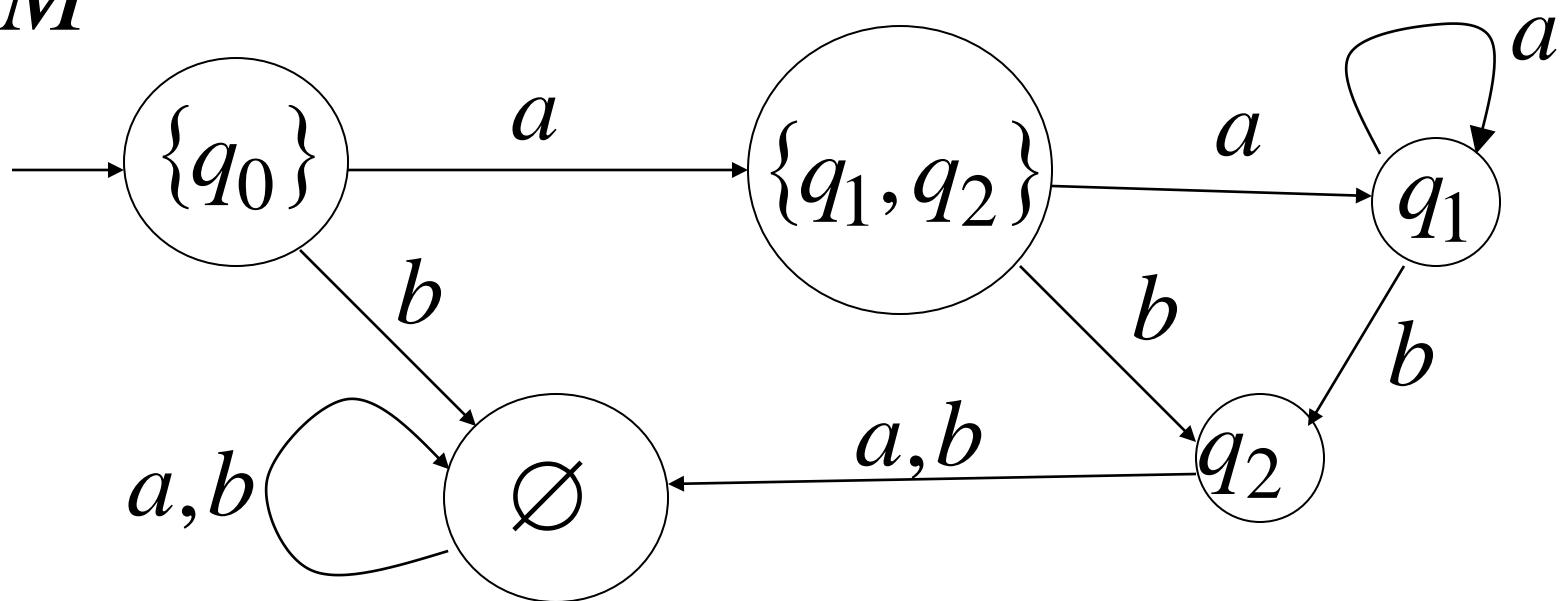


# Example

NFA  $M$



DFA  $M'$



step

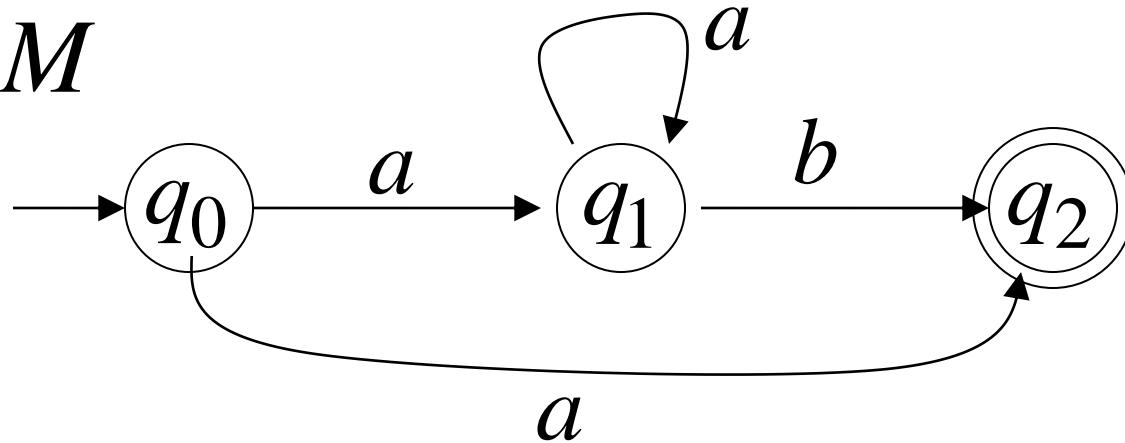
4. For any DFA state  $\{q_i, q_j, \dots, q_m\}$

if some  $q_j$  is accepting state in NFA

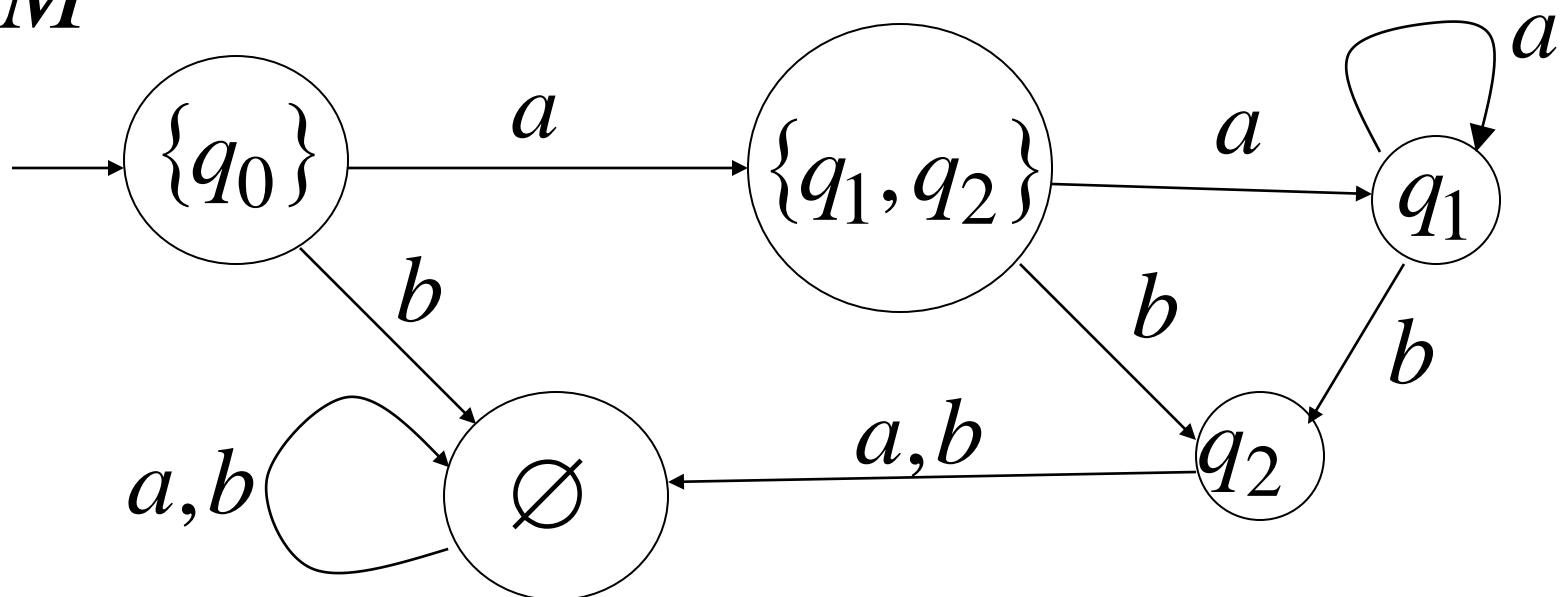
Then,  $\{q_i, q_j, \dots, q_m\}$   
is accepting state in DFA

# Example

NFA  $M$

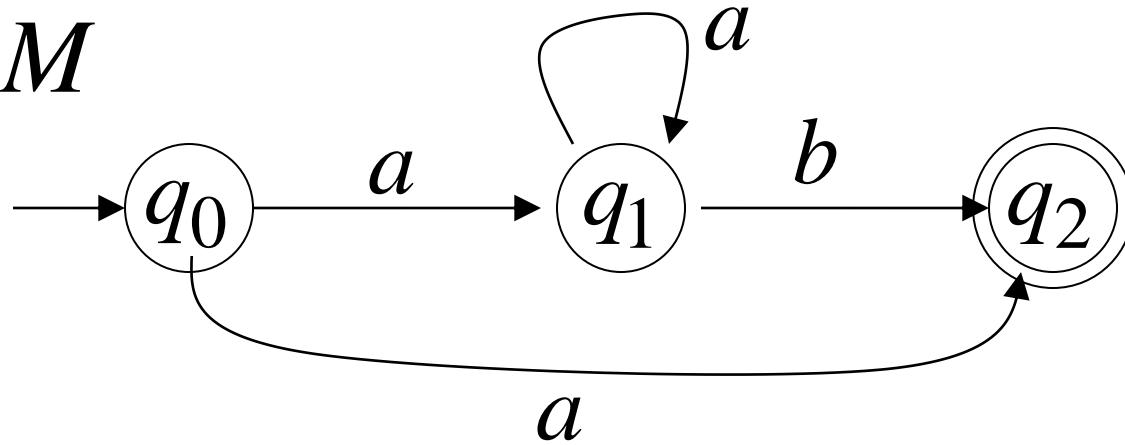


DFA  $M'$



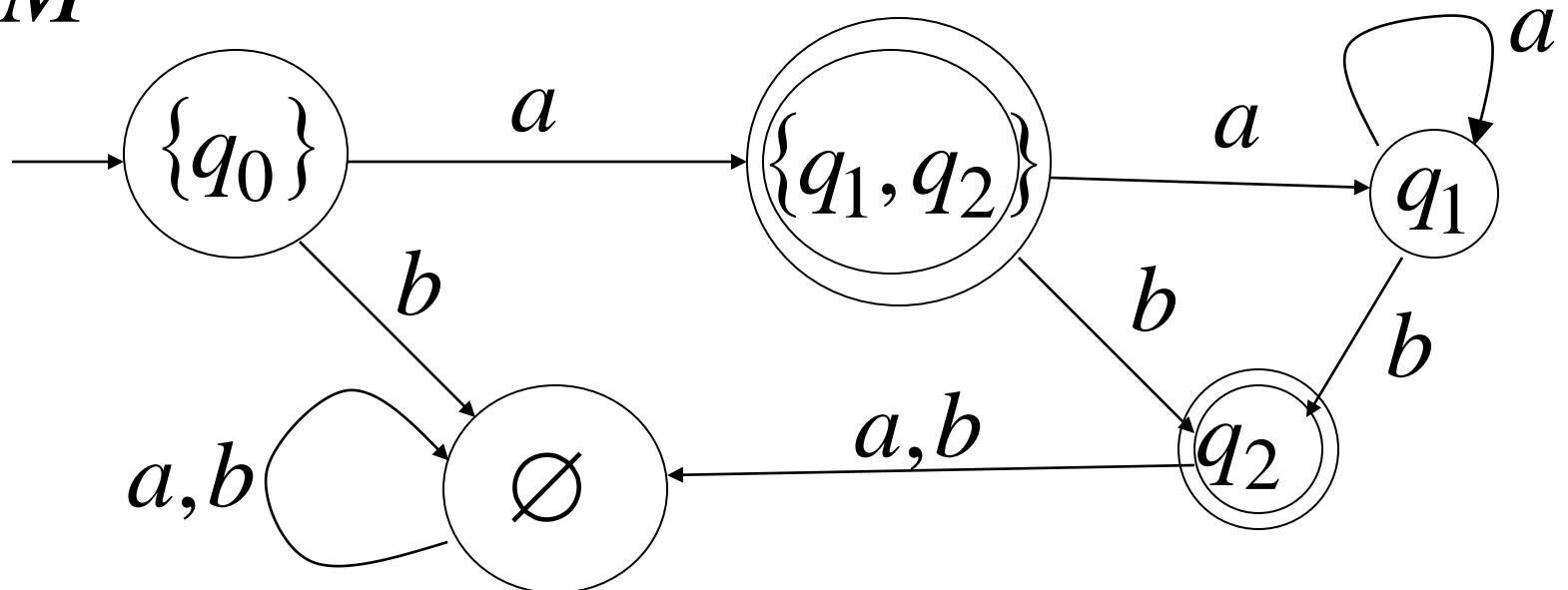
# Example

NFA  $M$



$$q_2 \in F$$

DFA  $M'$

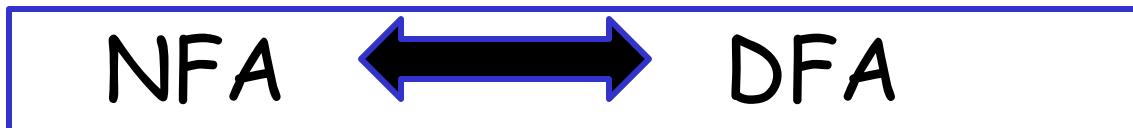
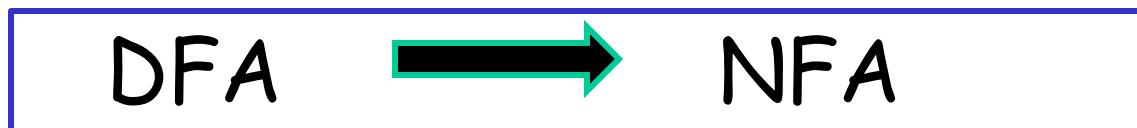
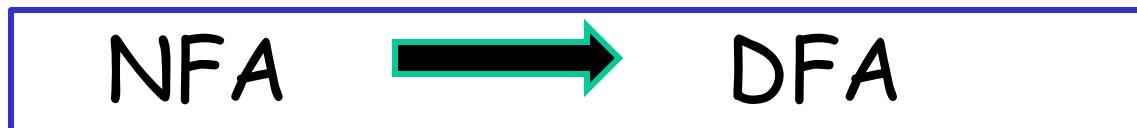


$$\{q_1, q_2\} \in F'$$

## Lemma:

If we convert NFA  $M$  to DFA  $M'$   
then the two automata are equivalent:

$$L(M) = L(M')$$



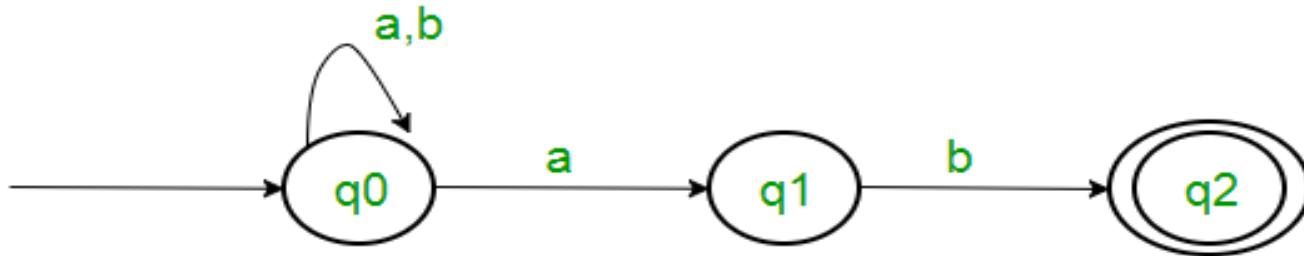
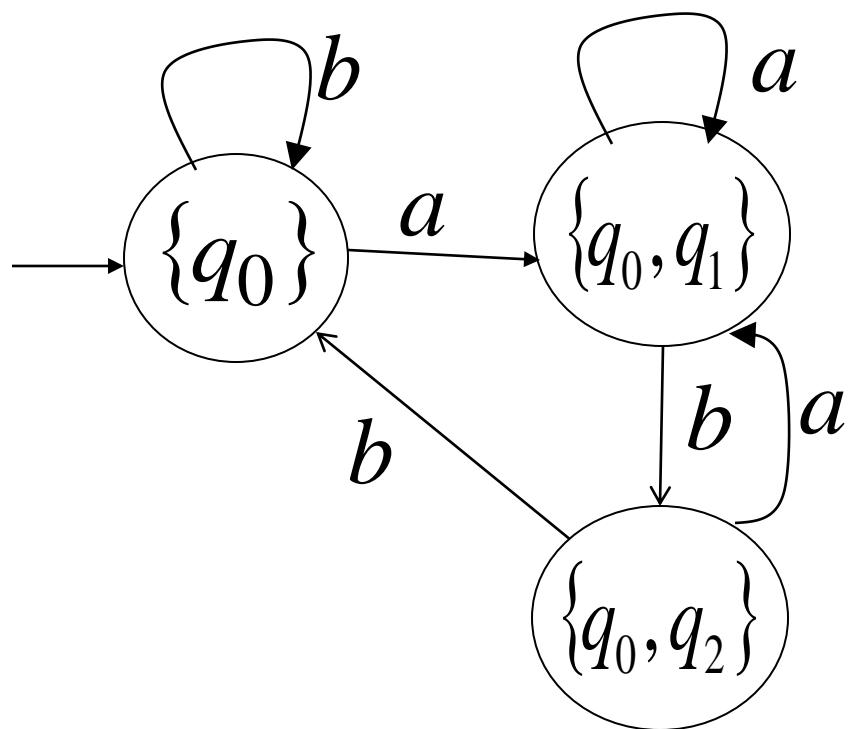


Figure 1

$$\delta^*(q_0, a) = \{q_0, q_1\}$$

$$\delta^*(\{q_0, q_1\}, a) = \delta^*(q_0, a) \cup \delta^*(q_1, a) = \{q_0, q_1\} \cup \Theta = \{q_0, q_1\}$$

$$\delta^*(\{q_0, q_1\}, b) = \delta^*(q_0, b) \cup \delta^*(q_1, b) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$$



$$\delta^*(\{q_0, q_2\}, a)$$

$$= \delta^*(q_0, a) \cup \delta^*(q_2, a)$$

$$= \{q_0, q_1\} \cup \{\Theta\} = \{q_0, q_1\}$$

$$\delta^*(\{q_0, q_2\}, b)$$

$$= \delta^*(q_0, b) \cup \delta^*(q_2, b)$$

$$= \{q_0\} \cup \{\Theta\} = \{q_0\}$$

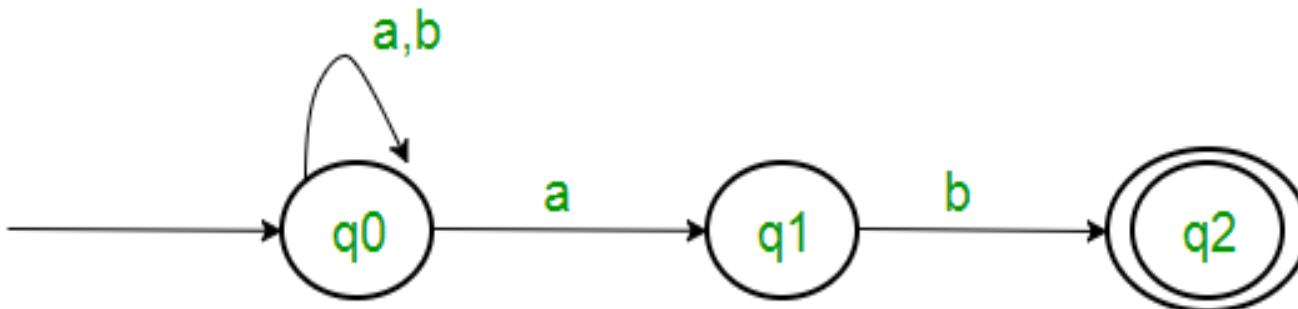
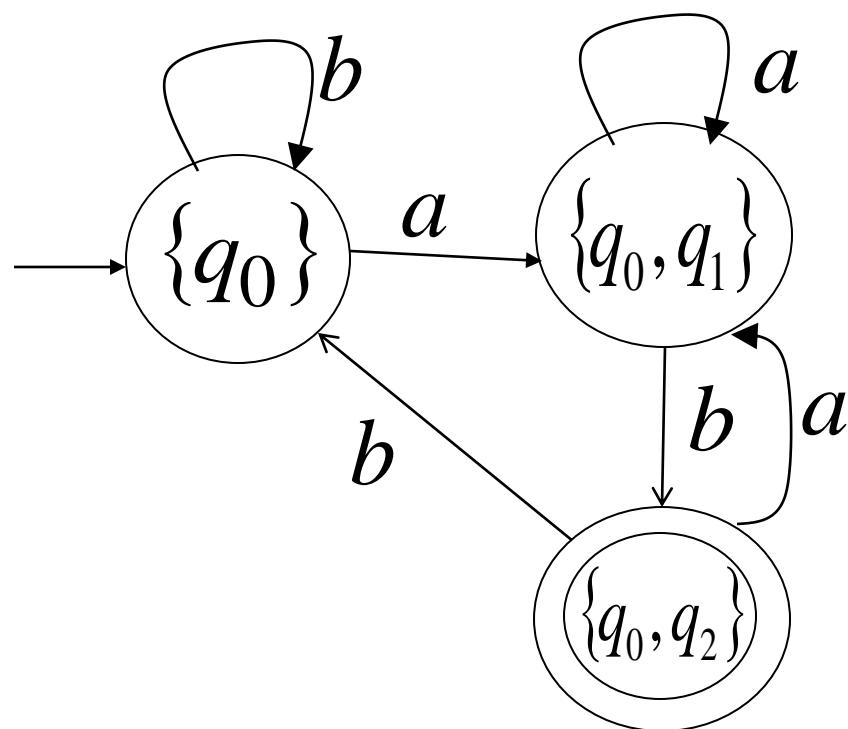


Figure 1



- Algorithm
- Input - An NDFA
- Output - An equivalent DFA
- Step 1 - Create state table from the given NDFA.
- Step 2 - Create a blank state table under possible input alphabets for the equivalent DFA.
- Step 3 - Mark the start state of the DFA by  $q_0$  (Same as the NDFA).
- Step 4 - Find out the combination of States  $\{Q_0, Q_1, \dots, Q_n\}$  for each possible input alphabet.
- Step 5 - Each time we generate a new DFA state under the input alphabet columns, we have to apply step 4 again, otherwise go to step 6.
- Step 6 - The states which contain any of the final states of the NDFA are the final states of the equivalent DFA.

# NFA

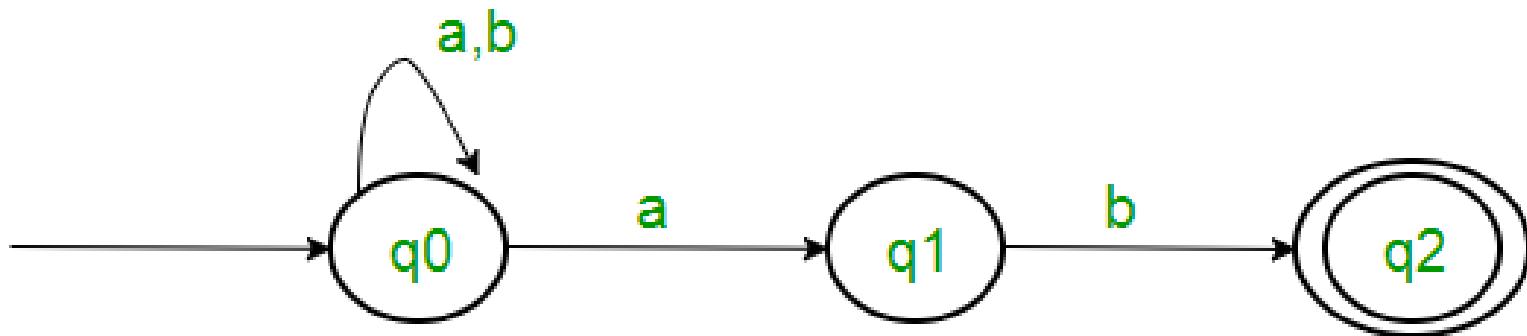


Figure 1

$$Q = \{q_0, q_1, q_2\}$$

$$\Sigma = a, b$$

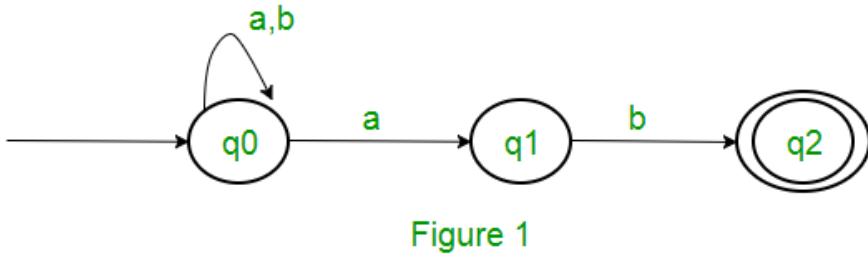
$$q_0 = q_0$$

$$F = \{q_2\}$$

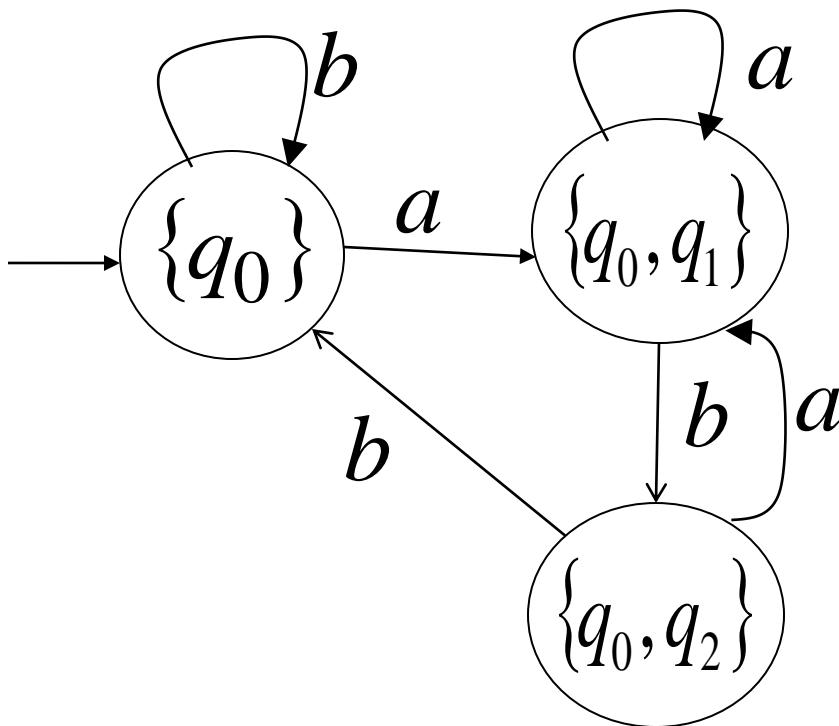
$\delta$  (Transition Function)

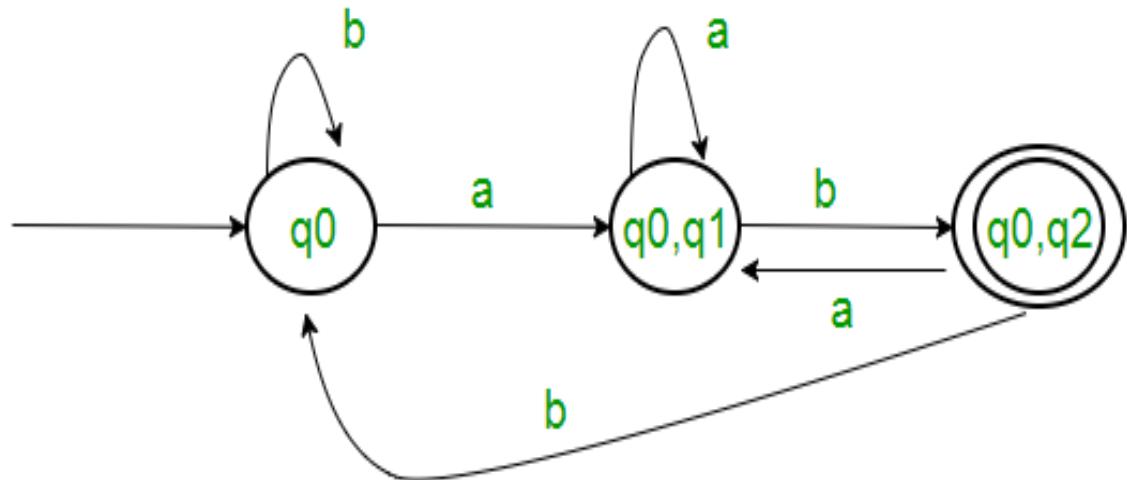
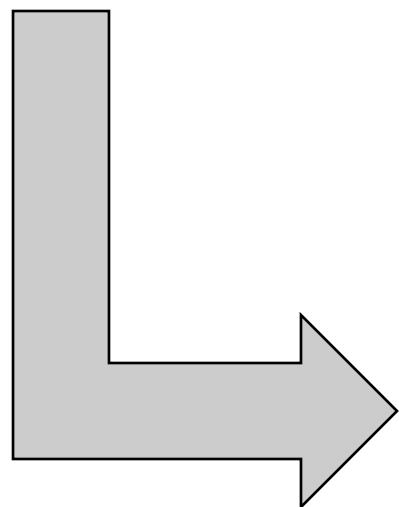
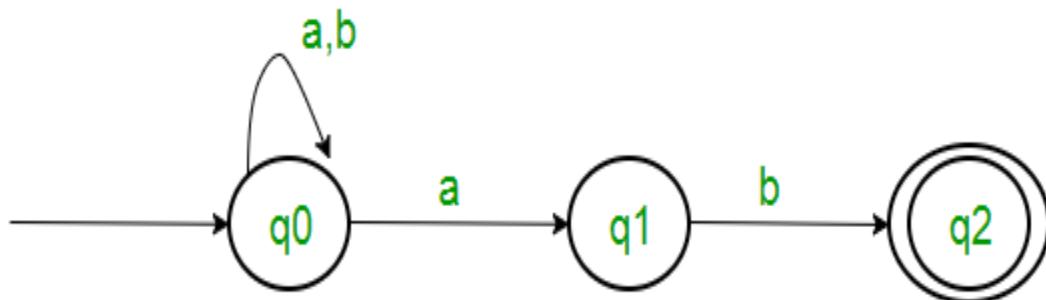
	$a$	$b$
$q_0$	$\{q_0, q_1\}$	$q_0$
$q_1$		$q_2$
$q_2$		

	a	b
q0	{q0,q1}	q0
q1		q2
q2		



State	a	B
q0	{q0,q1}	q0
{q0,q1}	{q0,q1}	{q0,q2}
{q0,q2}	{q0,q1}	q0





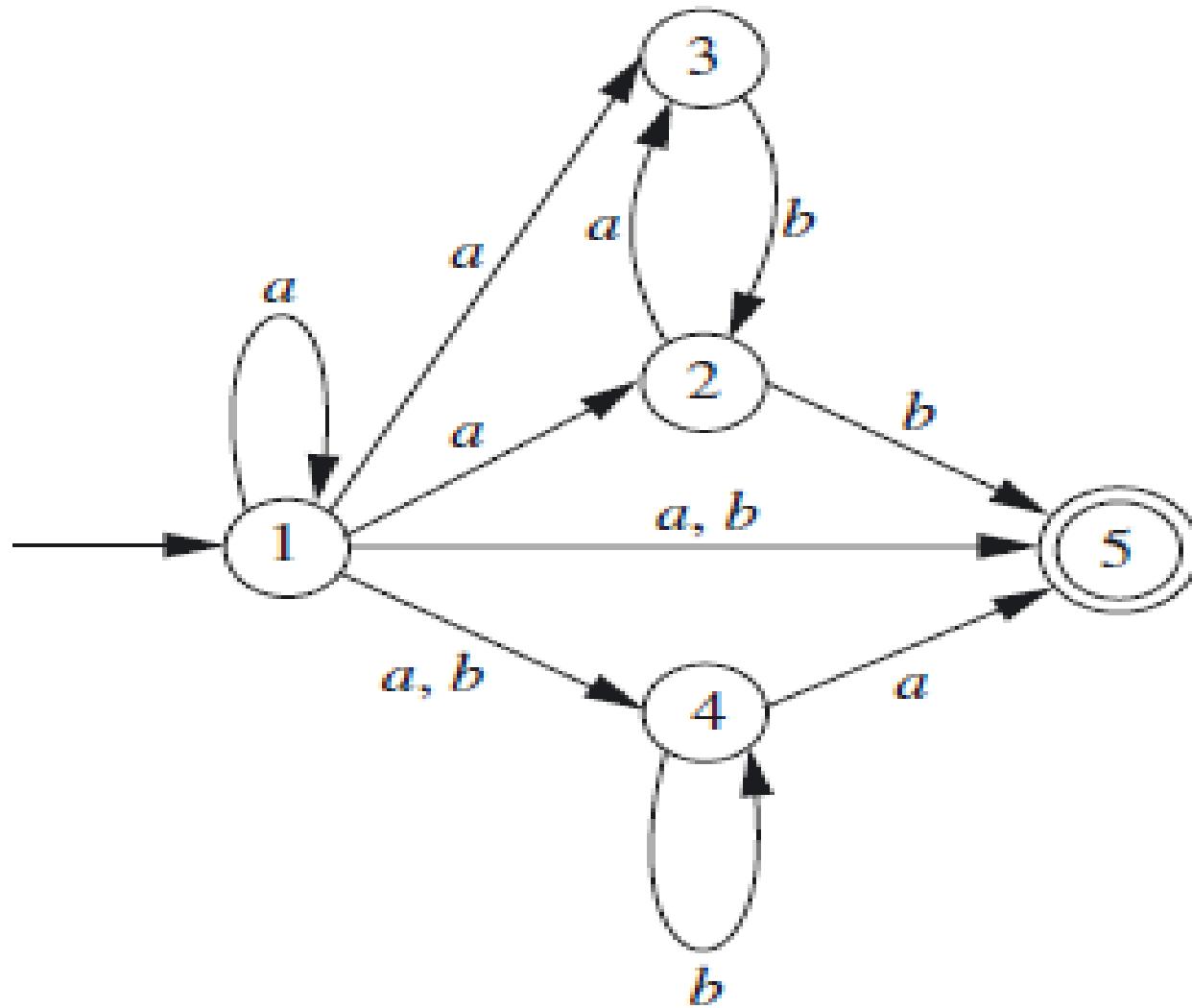
29/06/2019 Absent

- 5, 7, 9, 13, 16, 17, 21, 22, 26, 28, 32, 33, 34, 38, 39, 40, 41, 42, 43, 45, 46, 48, 51, 2, 4, 7, 9, 62, 5, 71, 72, 73, 74

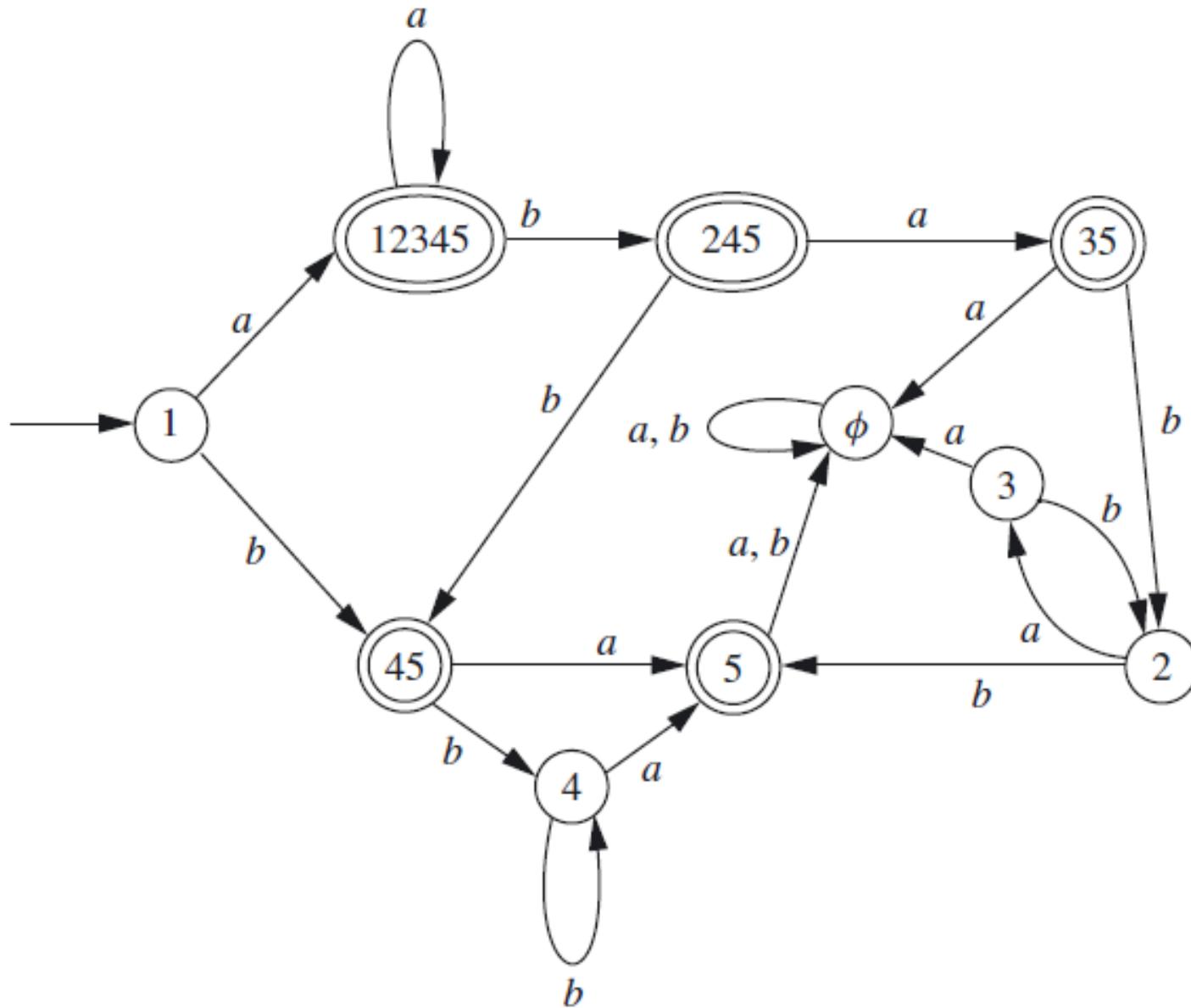
# RECALL

- Minimization of DFA
- DFA,NFA,NFA with Null accept regular language and Vice versa.
- DFA,NFA and NFA null Equivalent to each other.
- $\text{NFA} \rightarrow \text{DFA}$

# Example for NFA to DFA Conversion



# Output



# RECALL

- Minimization of DFA
- DFA,NFA,NFA with Null accept regular language and Vice versa.
- DFA,NFA and NFA null Equivalent to each other.
- $\text{NFA} \rightarrow \text{DFA}$
- $\text{NFA-null} \rightarrow \text{NFA}$
- $\text{NFA-null} \rightarrow \text{DFA}$

# NFA- $\lambda$ to NFA Conversion

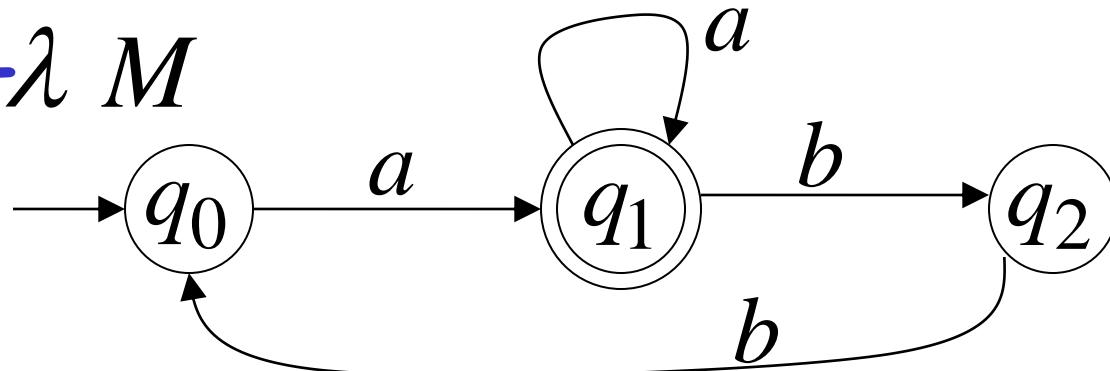
# General Conversion Procedure

Input: an NFA- $\lambda$   $M$

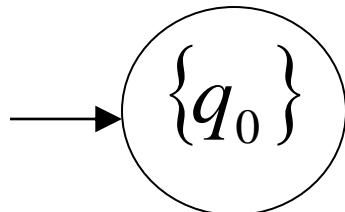
Output: an equivalent NFA  $M'$   
with  $L(M) = L(M')$

# Conversion NFA- $\lambda$ to NFA

NFA- $\lambda$   $M$



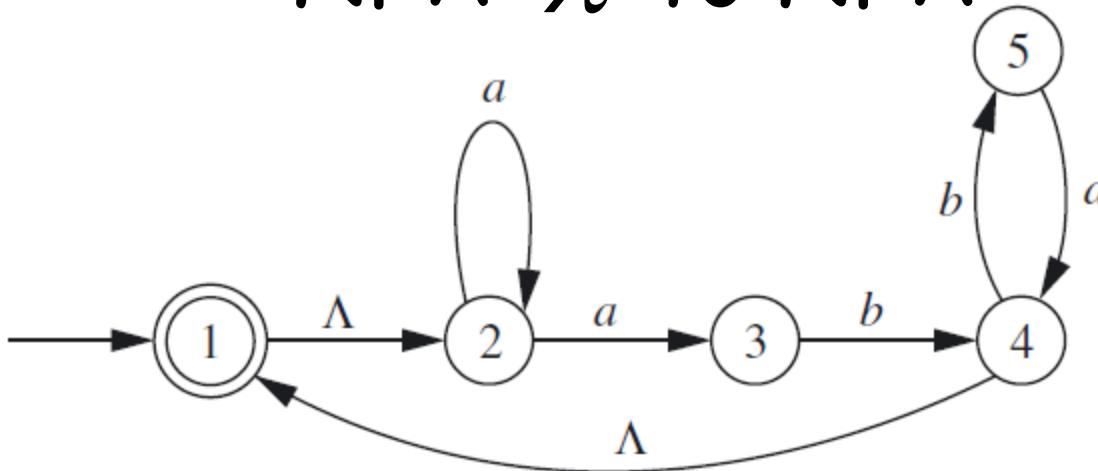
NFA  $M'$



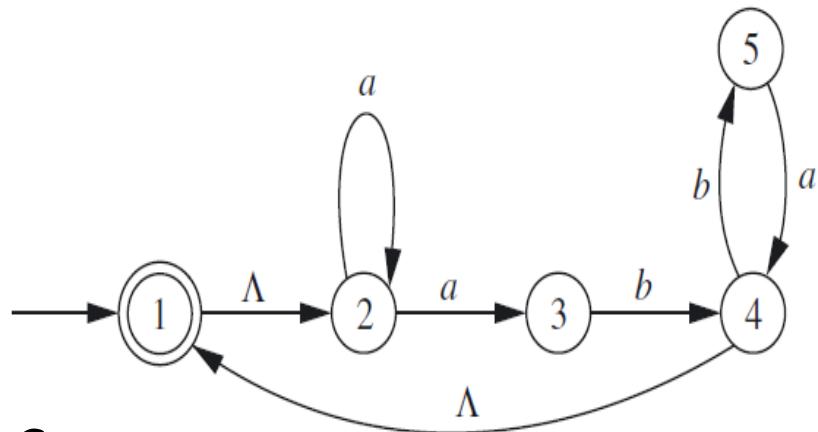
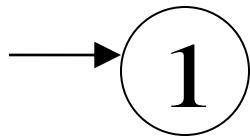
# The $\lambda$ -Closure of a Set of States

- Suppose  $M = (Q, \lambda, q_0, A, \delta)$  is an NFA, and  $S \subseteq Q$  is a set of states.
- The  $\lambda$ -closure of  $S$  is the set  $(S)$  that can be defined recursively as follows.
  1.  $S \subseteq (S)$ .
  2. For every  $q \in (S)$ ,  $\delta(q, \lambda) \subseteq (S)$ .

# NFA- $\lambda$ to NFA



1. States of NFA- $\lambda$  and NFA are Same
2. Initial state of NFA- $\lambda$  is initial state of NFA.
3. For transitions,
  - find the null closure of the state
  - apply the symbol from the alphabet
  - find the null closure of a set states generated from the above step
4. Repeat the step No.4 for each state



1. States of NFA-  $\lambda$  and NFA are Same
2. Initial state of NFA-  $\lambda$  is initial state of NFA.
3. For transitions,
  - find the null closure of the state
  - apply the symbol from the alphabet
  - find the null closure of the states generated from the above step

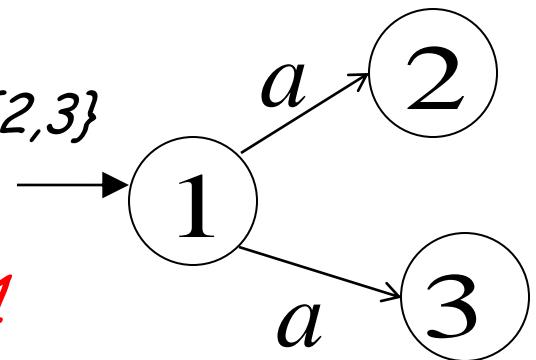
$$1) \lambda - (\{1\}) = \{1, 2\}$$

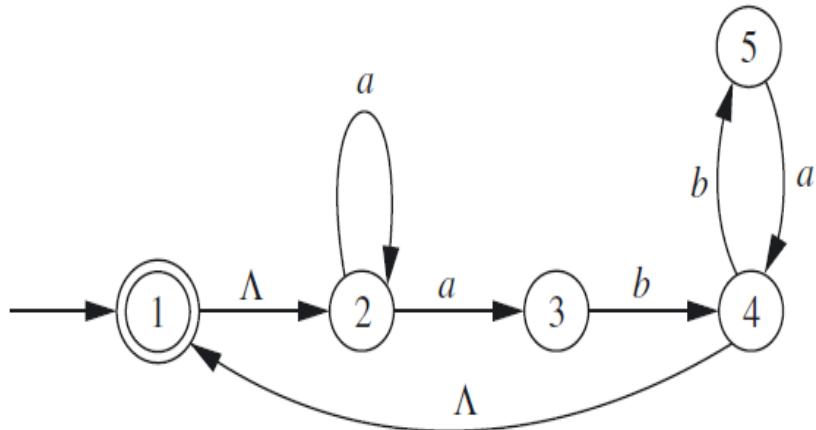
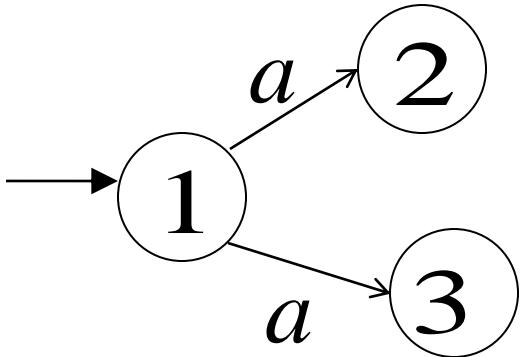
$$2) \delta(\{1, 2\}, a) = \delta(1, a) \cup \delta(2, a) = \emptyset \cup \{2, 3\} = \{2, 3\}$$

$$3) \lambda - \{2, 3\} = \{2, 3\}$$

4) After applying transitions  $a$  on state 1

It will move to state  $\{2, 3\}$





$$1) \lambda - \{1\} = \{1, 2\}$$

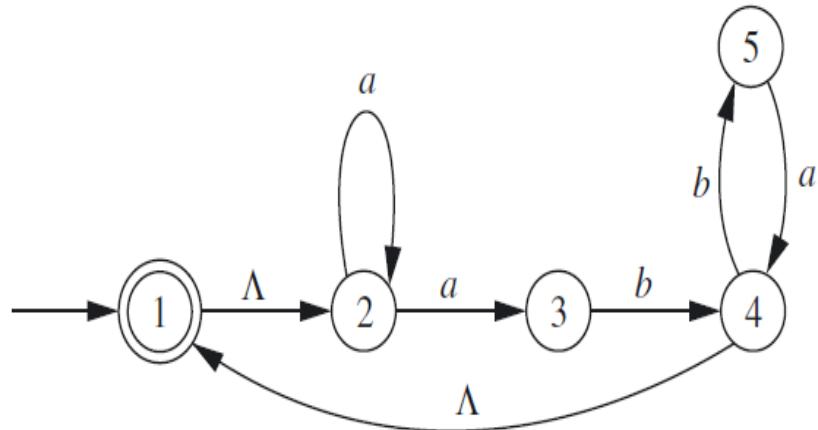
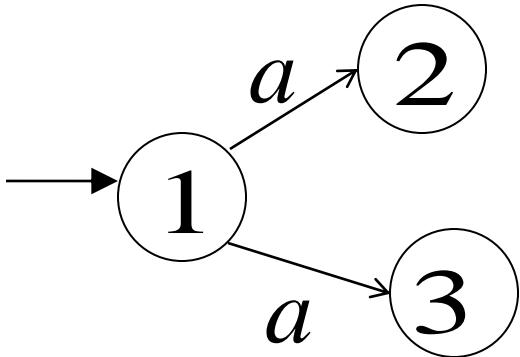
$$2) \delta(\{1, 2\}, a) = \delta(1, a) \cup \delta(2, a) = \emptyset \cup \{2, 3\}$$

$$3) \lambda - \{2, 3\} = \{2, 3\}$$

4) After applying transitions **a** on state **1**

It will move to state  $\{2, 3\}$

		<b>a</b>			<b>b</b>	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1, 2\}$	$\{2, 3\}$	$\{2, 3\}$			



$$1) \lambda - (\{1\}) = \{1, 2\}$$

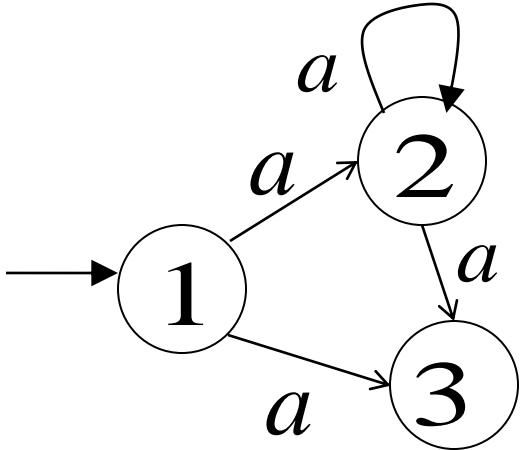
$$2) \delta(\{1, 2\}, b) = \delta(1, b) \cup \delta(2, b) = \emptyset \cup \emptyset$$

$$3) \lambda - \emptyset = \emptyset$$

4) After applying transitions **b** on state 1

It will move to the dead state

		a			b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2\}$	$\emptyset$	$\emptyset$



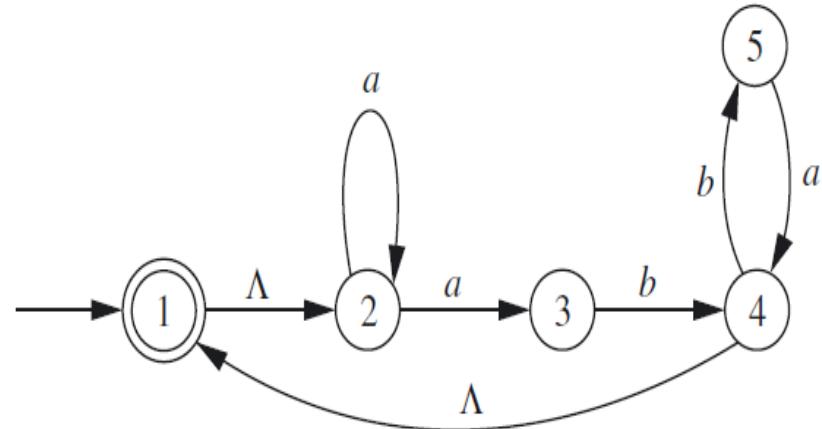
$$1) \lambda - (\{2\}) = \{2\}$$

$$2) \delta(\{2\}, a) = \{2, 3\}$$

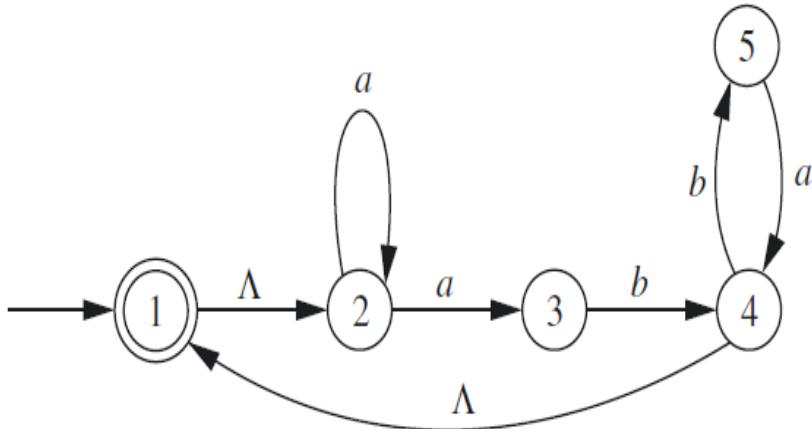
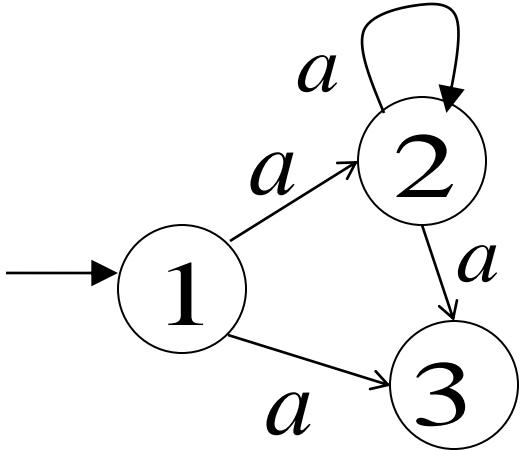
$$3) \lambda - \{2, 3\} = \{2, 3\}$$

4) After applying transitions  $a$  on state 2

It will move to state  $\{2, 3\}$



		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2\}$	$\emptyset$	$\emptyset$
2	$\{2\}$	$\{2, 3\}$	$\{2, 3\}$			



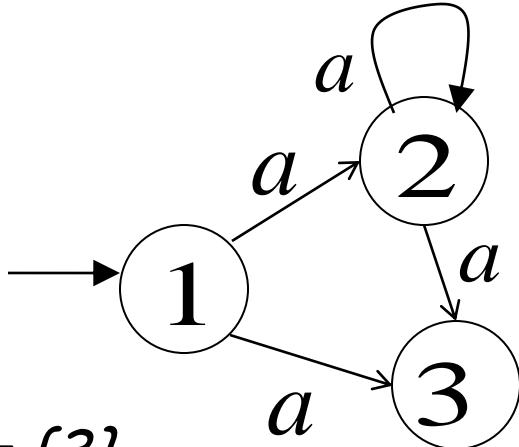
- 1)  $\lambda - (\{2\}) = \{2\}$
- 2)  $\delta(\{2\}, b) = \emptyset$

$$3) \lambda - \emptyset = \emptyset$$

4) After applying transitions  $b$  on state 2

It will move to Dead state.

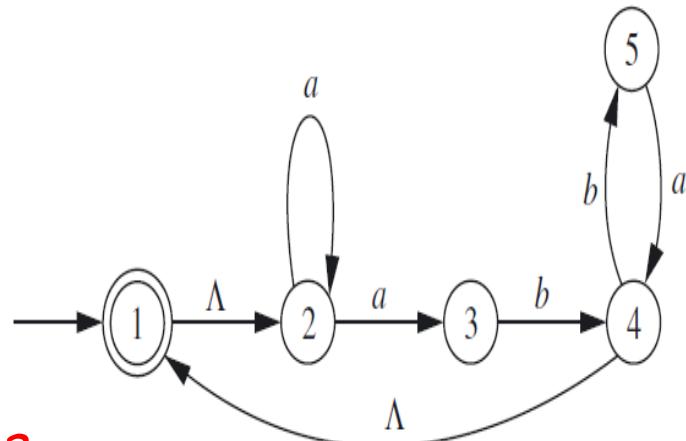
		a			b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2\}$	$\{2,3\}$	$\{2,3\}$	$\{1,2\}$	$\emptyset$	$\emptyset$
2	$\{2\}$	$\{2,3\}$	$\{2,3\}$	$\{2\}$	$\emptyset$	$\emptyset$



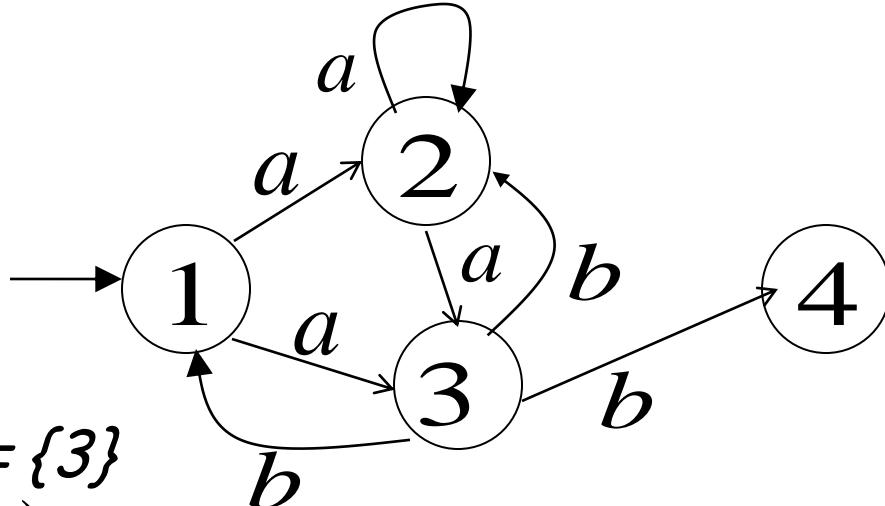
- 1)  $\lambda - (\{3\}) = \{3\}$
- 2)  $\delta(\{3\}, a) = \emptyset$
- 3)  $\lambda - \emptyset = \emptyset$

4) After applying transitions **a** on state 3

It will move to Dead State



		a			b	
1	$\{1,2\}$	$\{2,3\}$	$\{2,3\}$	$\{1,2\}$	$\emptyset$	$\lambda_{\emptyset}$
2	$\{2\}$	$\{2,3\}$	$\{2,3\}$	$\{2\}$	$\emptyset$	$\emptyset$
3	$\{3\}$	$\emptyset$	$\emptyset$			
4						
5						



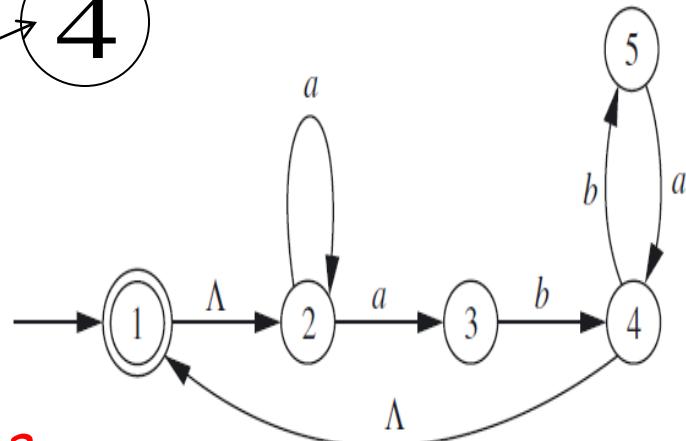
$$1) \lambda - \{3\} = \{3\}$$

$$2) \delta(\{3\}, b) = 4$$

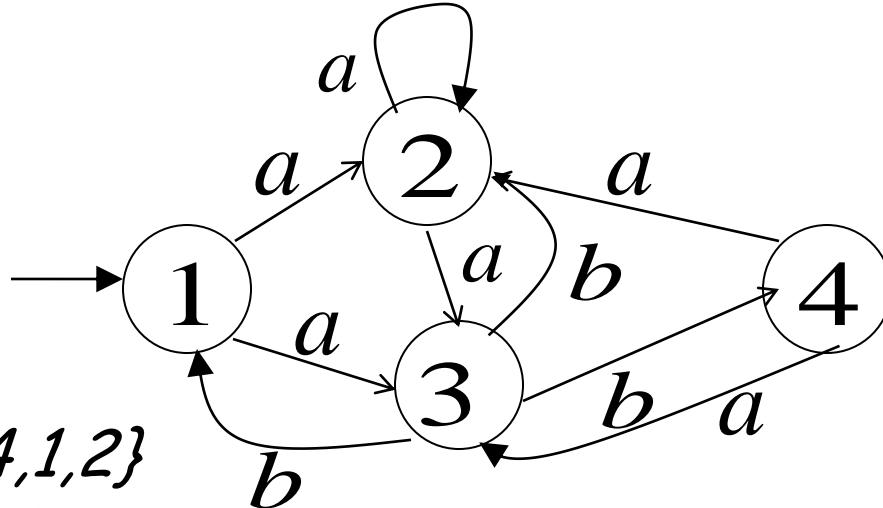
$$3) \lambda - \{4\} = \{4, 1, 2\}$$

4) After applying transitions  $b$  on state 3

It will move to State 1, 2 and 4



	a				b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2\}$	$\emptyset$	$\emptyset$
2	$\{2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{2\}$	$\emptyset$	$\emptyset$
3	$\{3\}$	$\emptyset$	$\emptyset$	$\{3\}$	$\{4\}$	$\{4, 1, 2\}$
4						
5						



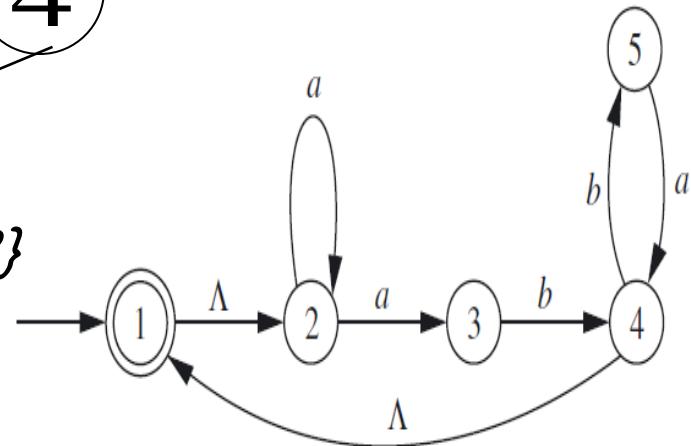
$$1) \lambda - \{4\} = \{4, 1, 2\}$$

$$2) \delta(\{4, 1, 2\}, a) = \delta(4, a) \cup \delta(1, a) \cup \delta(2, a) = \{2, 3\}$$

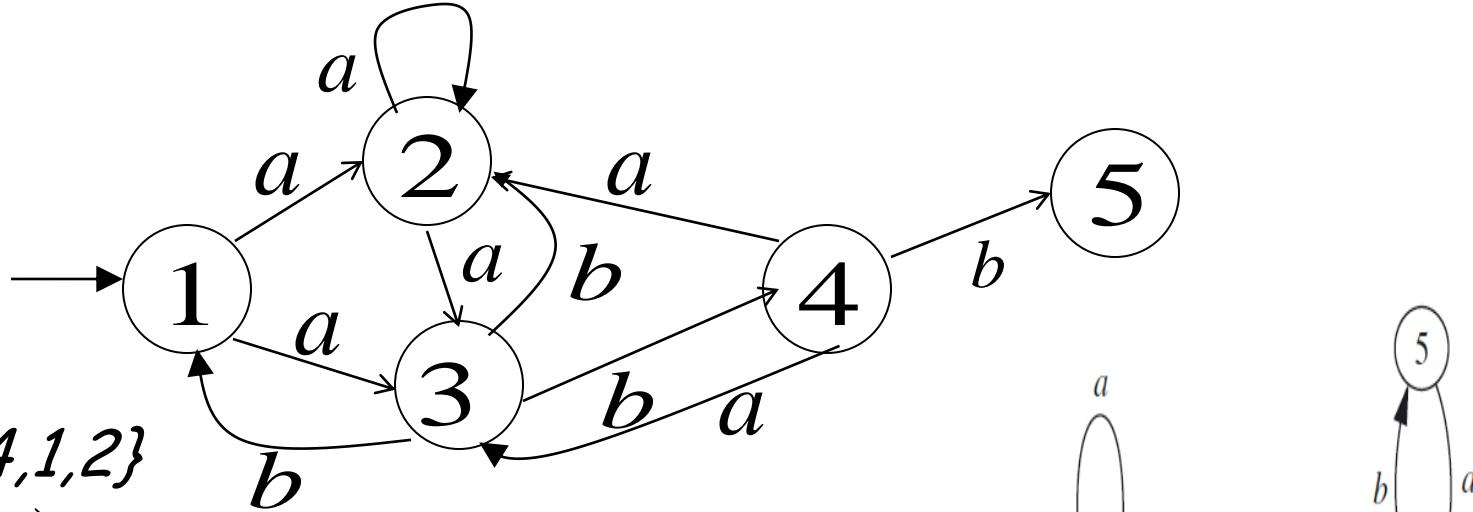
$$3) \lambda - \{2, 3\} = \{2, 3\}$$

4) After applying transitions **a** on state 4

It will move to State **2 and 3**



	<b>a</b>				<b>b</b>	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2\}$	$\emptyset$	$\emptyset$
2	$\{2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{2\}$	$\emptyset$	$\emptyset$
3	$\{3\}$	$\emptyset$	$\emptyset$	$\{3\}$	$\{4\}$	$\{4, 1, 2\}$
4	$\{4, 1, 2\}$	$\{2, 3\}$	$\{2, 3\}$			
5						



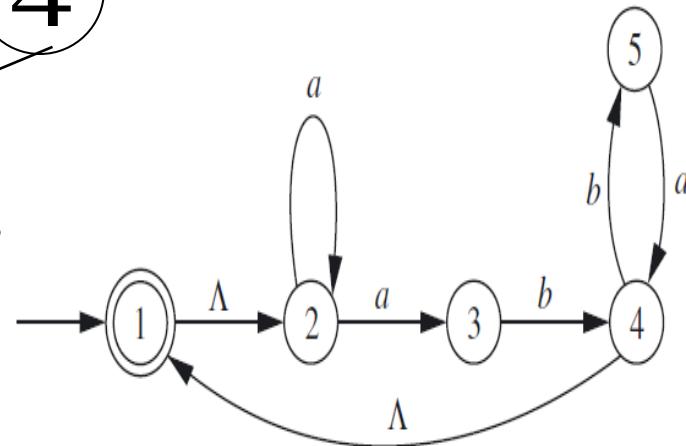
$$1) \lambda - \{4\} = \{4, 1, 2\}$$

$$2) \delta(\{4, 1, 2\}, b) = \delta(4, b) \cup \delta(1, b) \cup \delta(2, b) = \{5\}$$

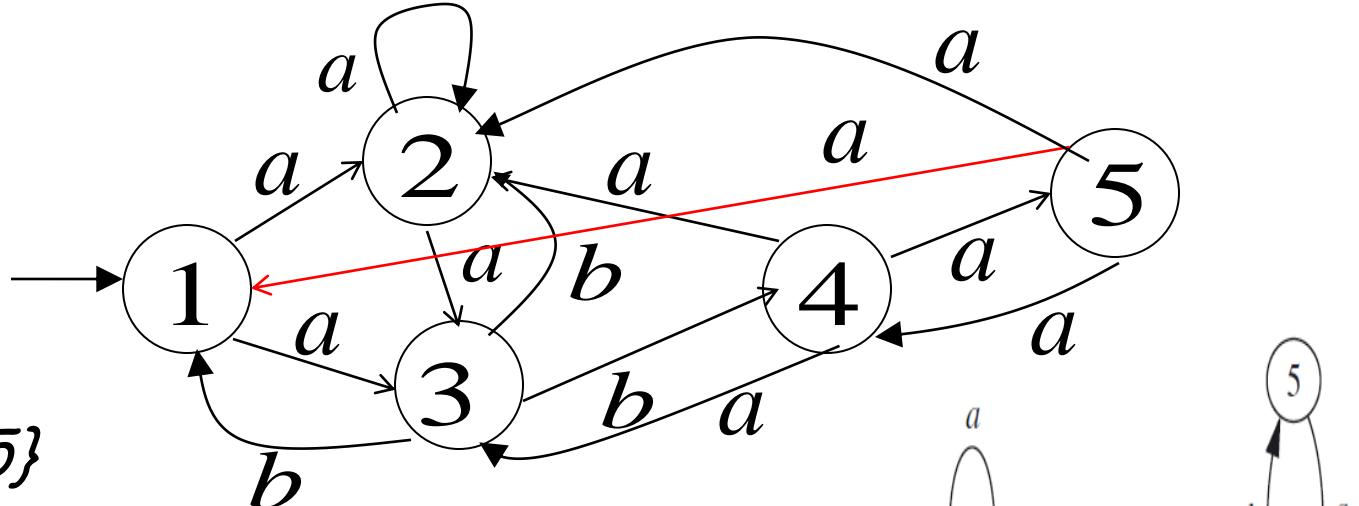
$$3) \lambda - \{5\} = \{5\}$$

4) After applying transitions  $b$  on state 4

It will move to State 5



	$a$				$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2\}$	$\emptyset$	$\emptyset$
2	$\{2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{2\}$	$\emptyset$	$\emptyset$
3	$\{3\}$	$\emptyset$	$\emptyset$	$\{3\}$	$\{4\}$	$\{4, 1, 2\}$
4	$\{4, 1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2, 4\}$	$\{5\}$	$\{5\}$
5						



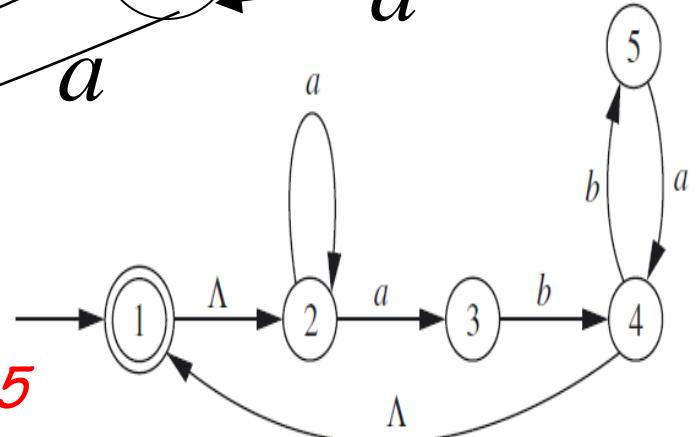
$$1) \lambda - \{5\} = \{5\}$$

$$2) \delta(\{5\}, a) = \{4\}$$

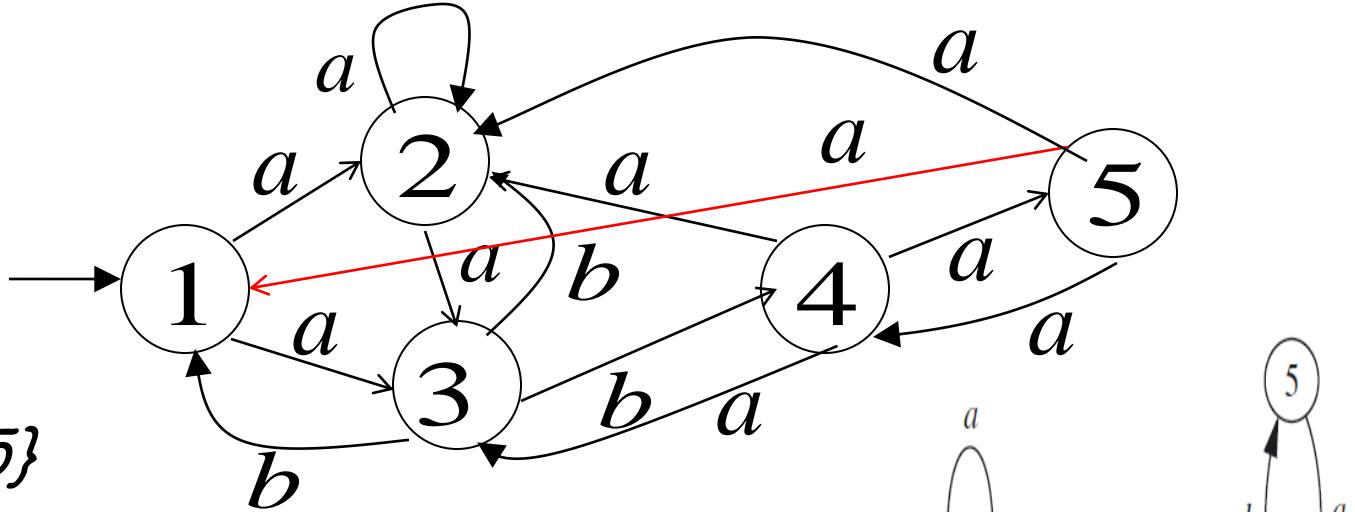
$$3) \lambda - \{4\} = \{1, 2, 4\}$$

4) After applying transitions  $a$  on state  $5$

It will move to State  $1, 2$  and  $4$



	$a$			$b$		
	$\lambda$		$\lambda$	$\lambda$		
1	$\{1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2\}$	$\emptyset$	$\emptyset$
2	$\{2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{2\}$	$\emptyset$	$\emptyset$
3	$\{3\}$	$\emptyset$	$\emptyset$	$\{3\}$	$\{4\}$	$\{4, 1, 2\}$
4	$\{4, 1, 2\}$	$\{2, 3\}$	$\{2, 3\}$	$\{1, 2, 4\}$	$\{5\}$	$\{5\}$
5	$\{5\}$	$\{4\}$	$\{1, 2, 4\}$			



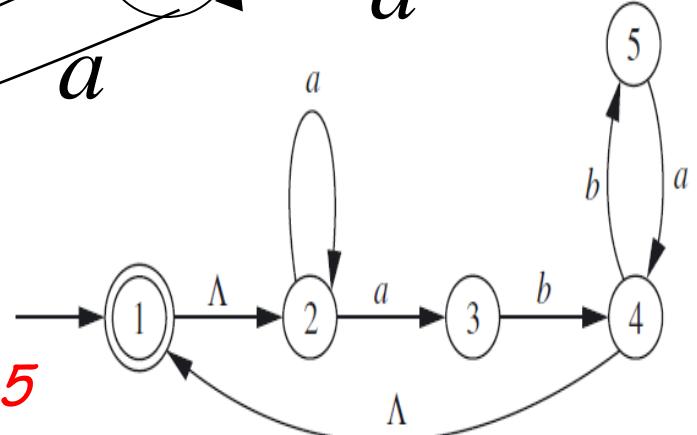
$$1) \lambda - \{5\} = \{5\}$$

$$2) \delta(\{5\}, b) = \{\emptyset\}$$

$$3) \lambda - \{\emptyset\} = \{\emptyset\}$$

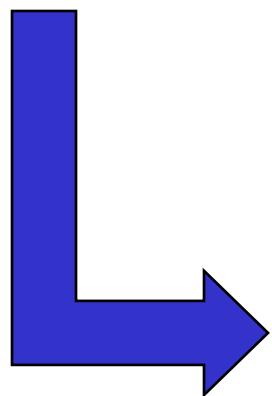
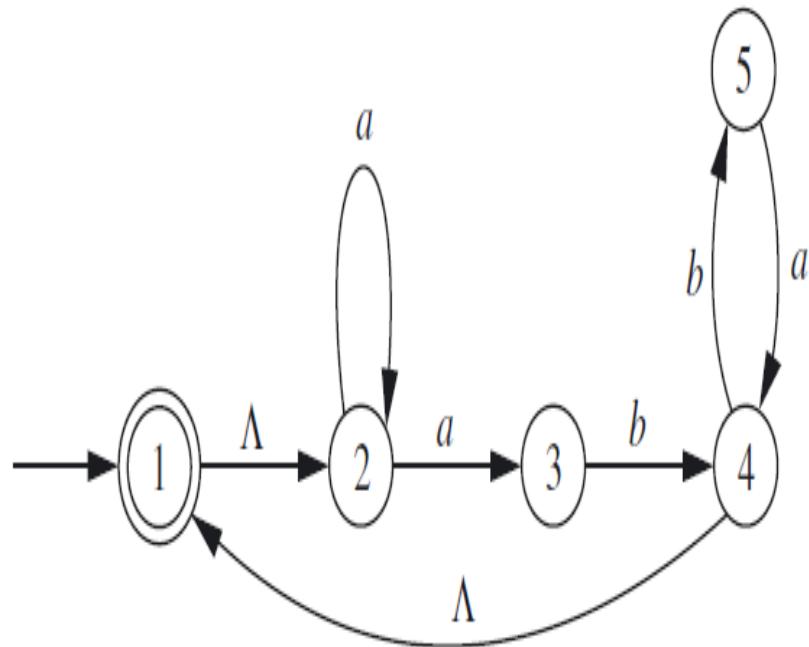
4) After applying transitions  $b$  on state 5

It will move to Dead State

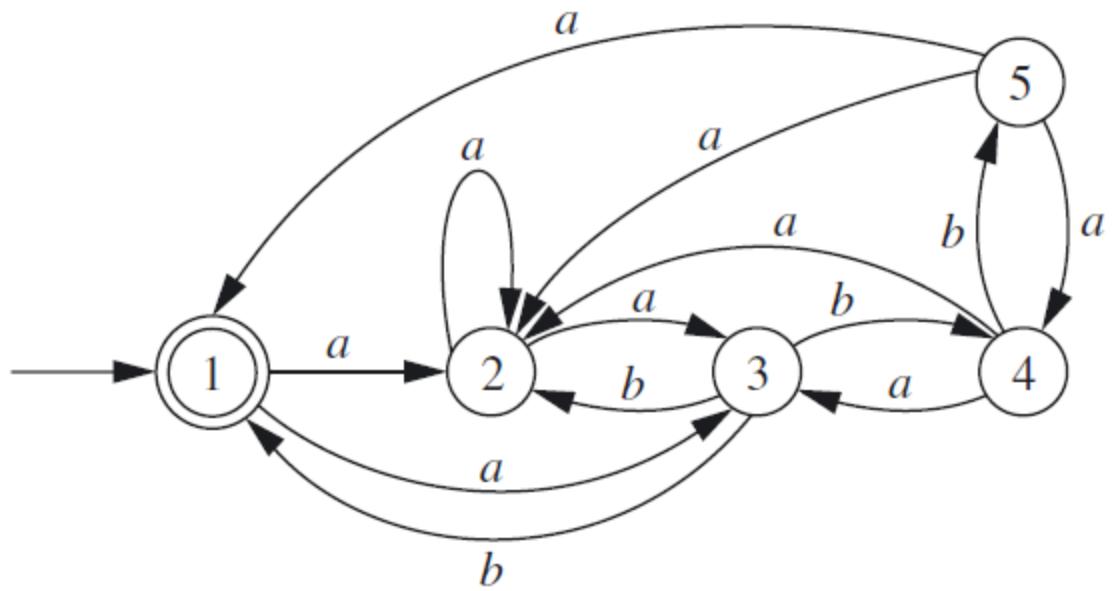


	a				b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2\}$	$\{2,3\}$	$\{2,3\}$	$\{1,2\}$	$\emptyset$	$\emptyset$
2	$\{2\}$	$\{2,3\}$	$\{2,3\}$	$\{2\}$	$\emptyset$	$\emptyset$
3	$\{3\}$	$\emptyset$	$\emptyset$	$\{3\}$	$\{4\}$	$\{4,1,2\}$
4	$\{4,1,2\}$	$\{2,3\}$	$\{2,3\}$	$\{1,2,4\}$	$\{5\}$	$\{5\}$
5	$\{5\}$	$\{4\}$	$\{1,2,4\}$	$\{5\}$	$\emptyset$	$\emptyset$

NFA- $\lambda$

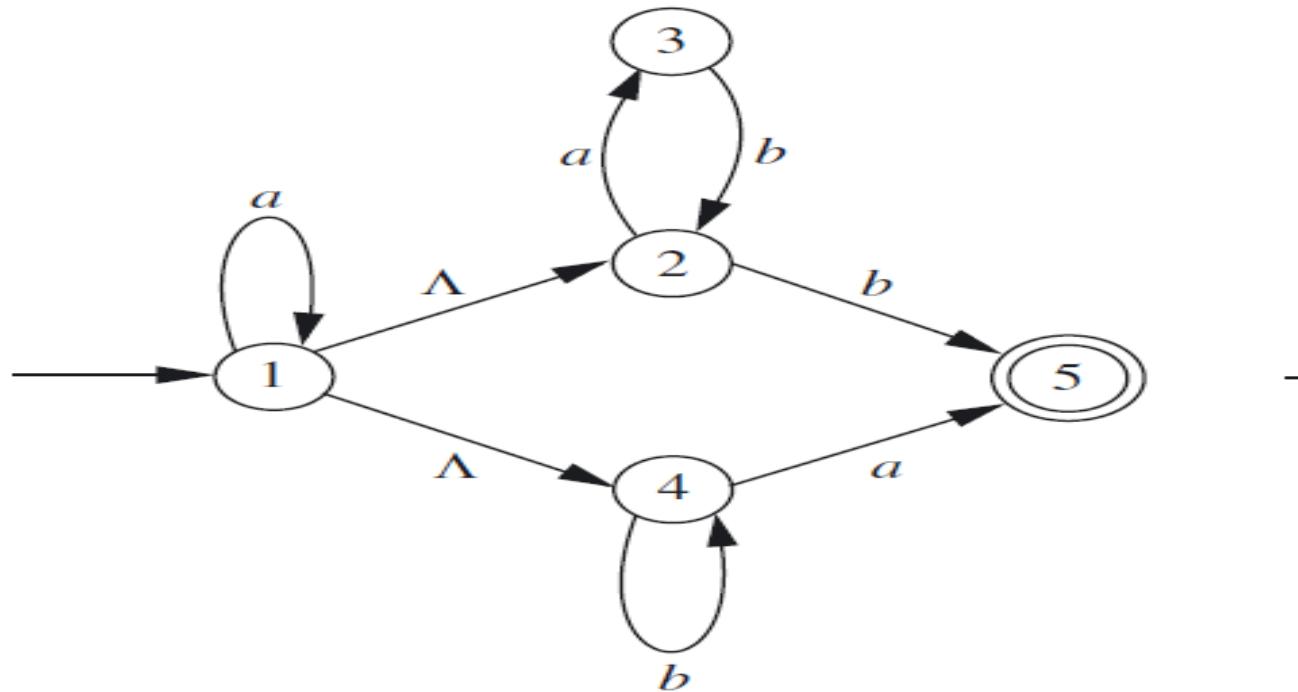


NFA

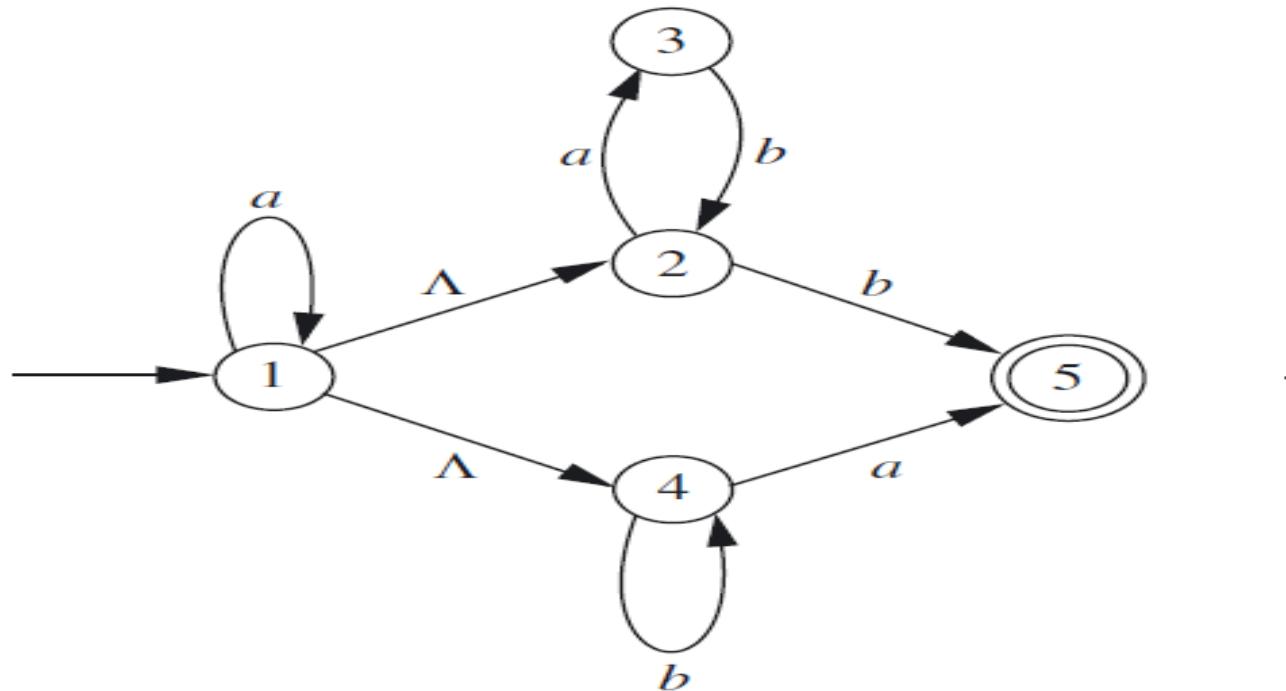


# RECALL

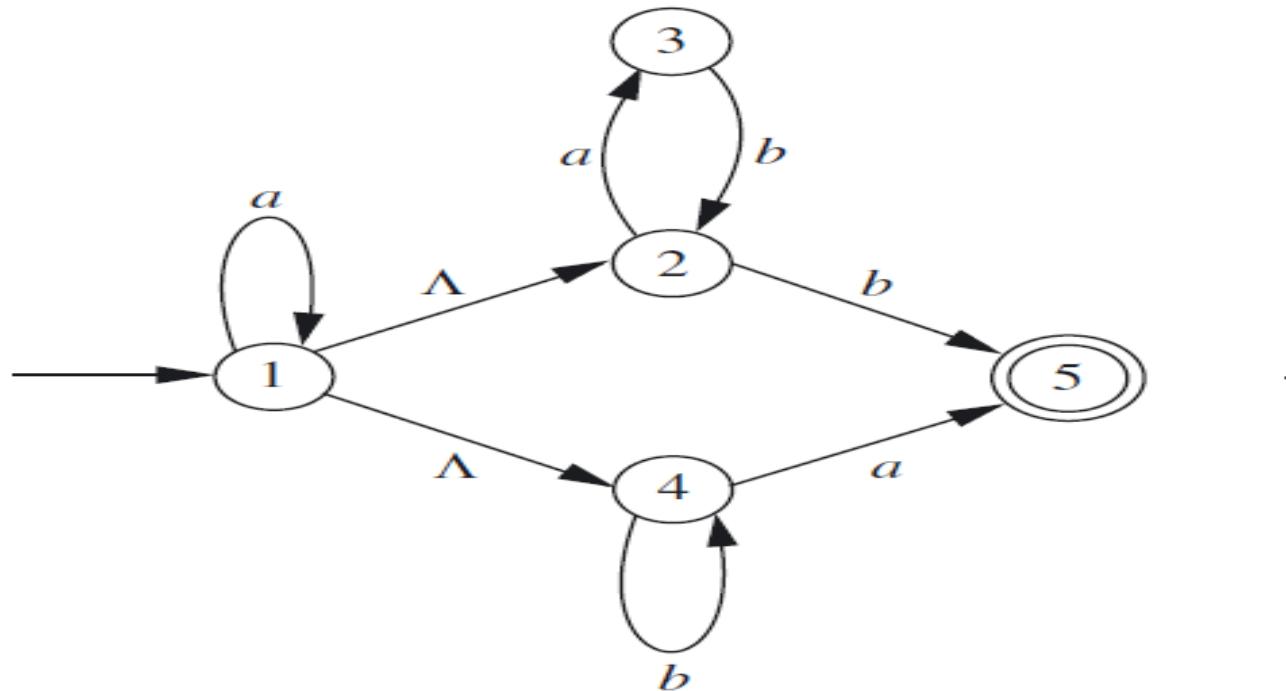
- Minimization of DFA
- DFA,NFA,NFA with Null accept regular language and Vice versa.
- DFA,NFA and NFA null Equivalent to each other.
- $\text{NFA} \rightarrow \text{DFA}$
- $\text{NFA-null} \rightarrow \text{NFA}$
- $\text{NFA-null} \rightarrow \text{DFA}$



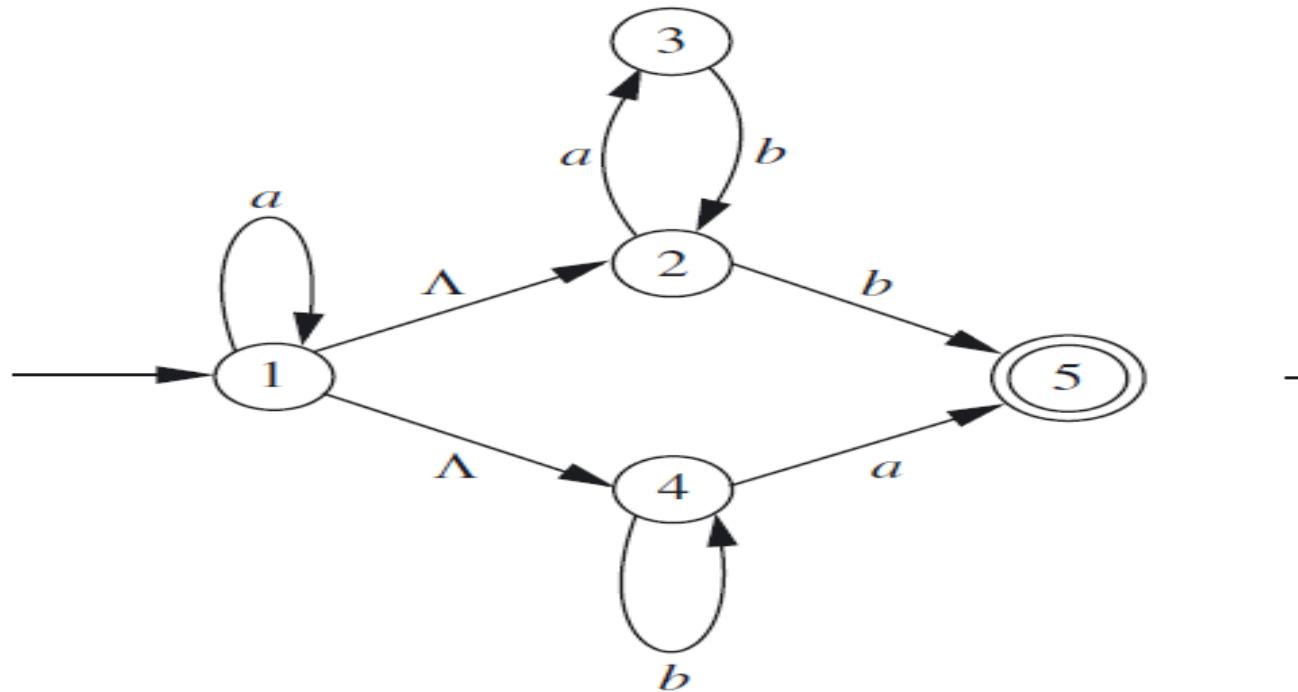
	$\lambda$	$a$			$b$	
1	$\lambda$		$\lambda$	$\lambda$		$\lambda$
2						
3						
4						
5						



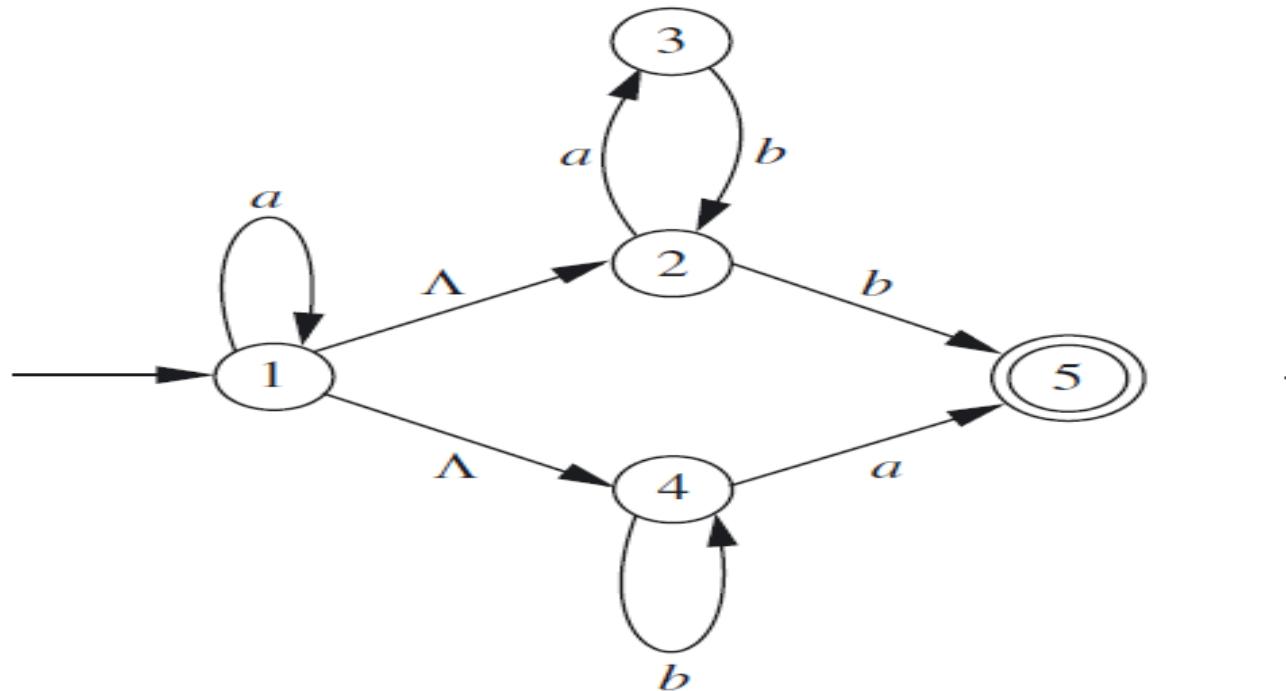
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,		$\lambda$	$\lambda$		$\lambda$
2						
3						
4						
5						



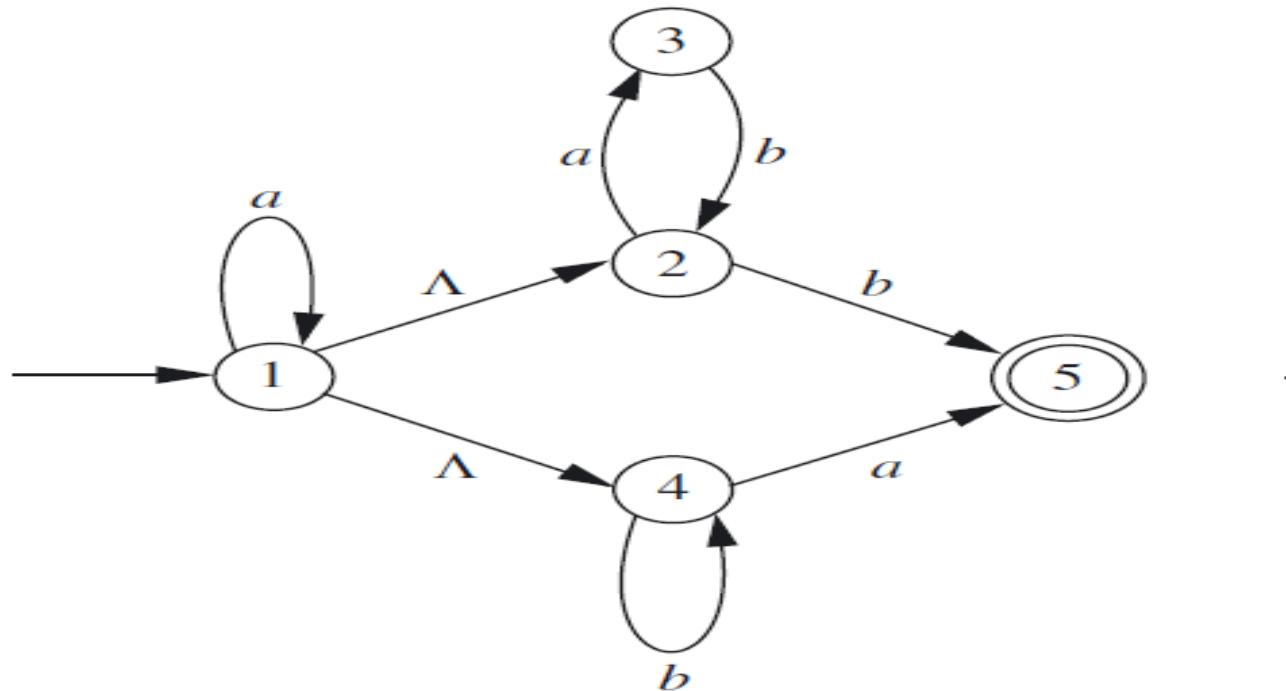
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\lambda$	$\lambda$	$\lambda$	$\lambda$
2						
3						
4						
5						



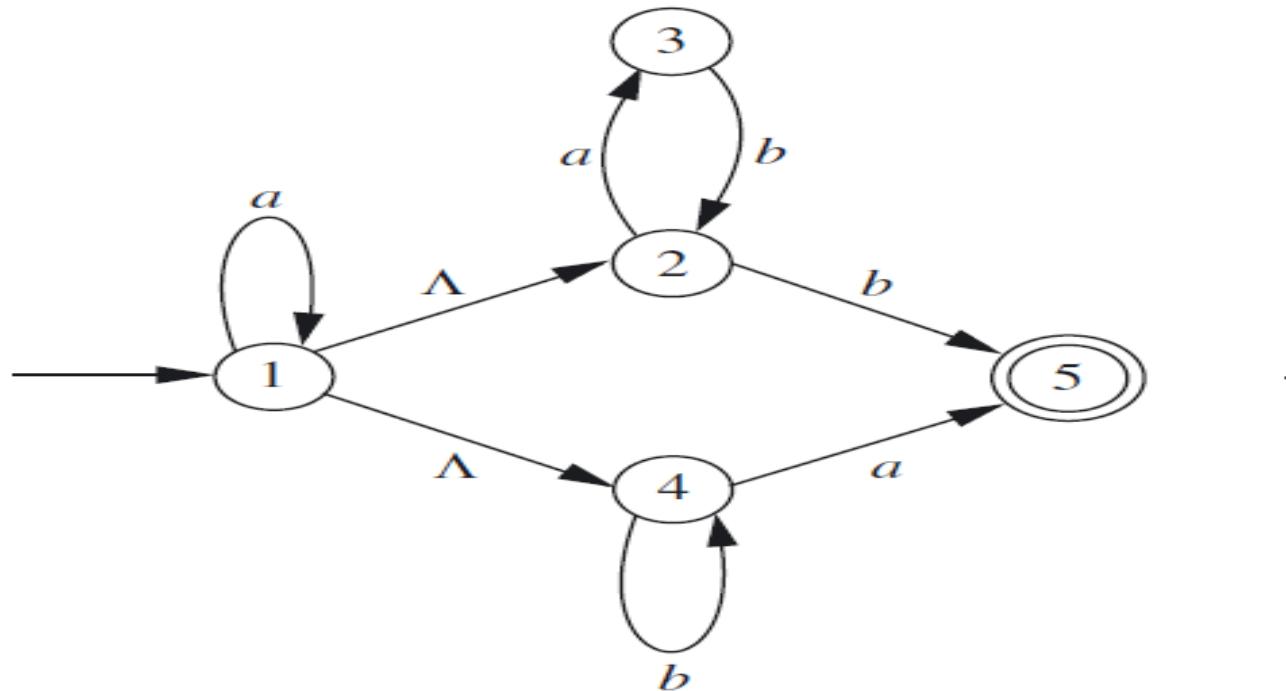
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$			$\lambda$
2						
3						
4						
5						



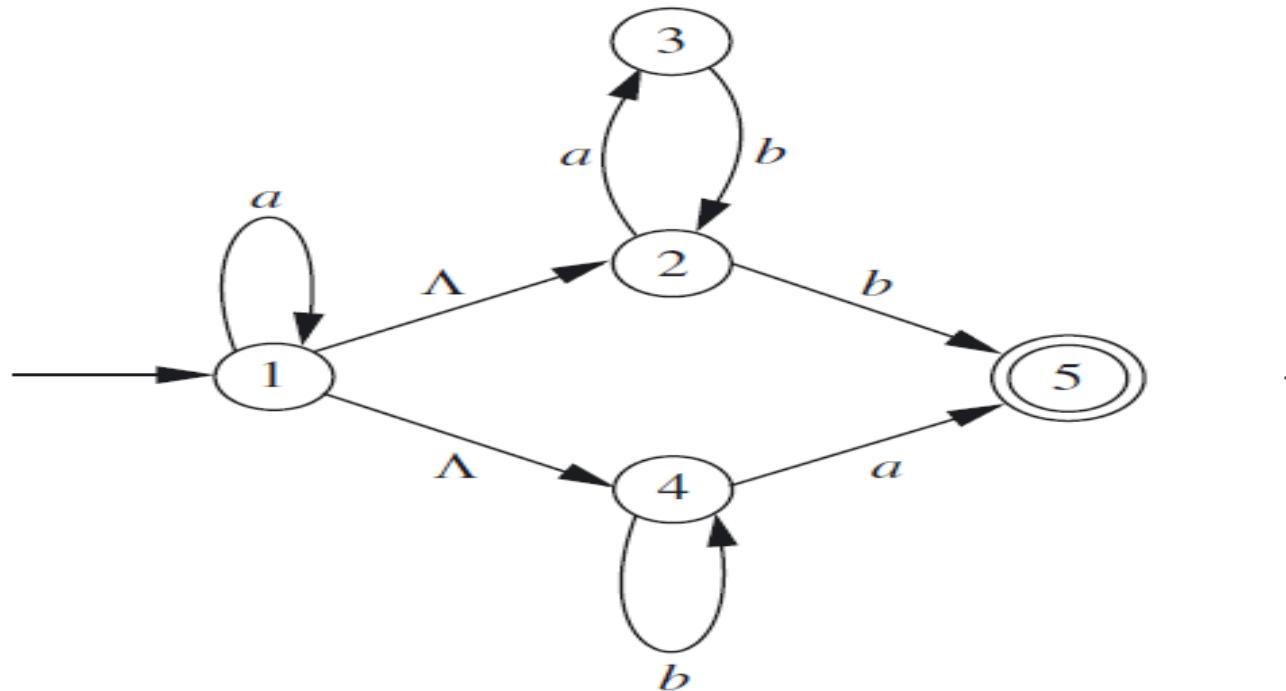
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{\lambda\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{\lambda\}$
2						
3						
4						
5						



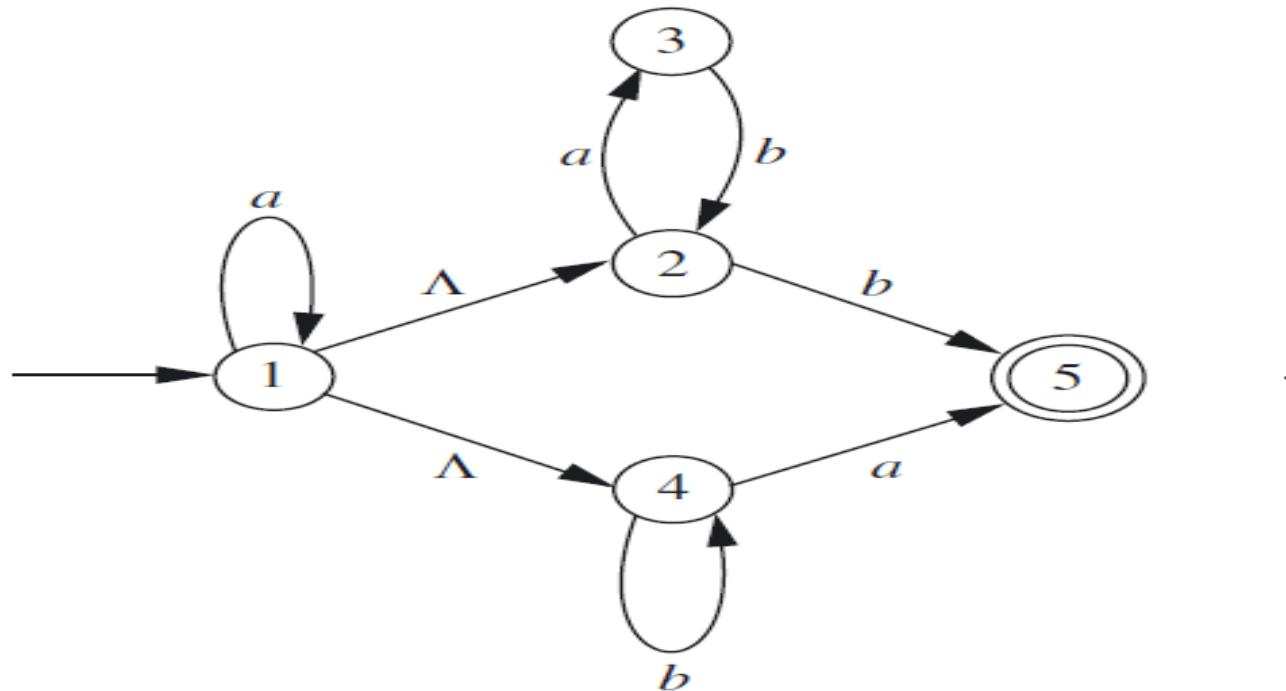
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	
2						
3						
4						
5						



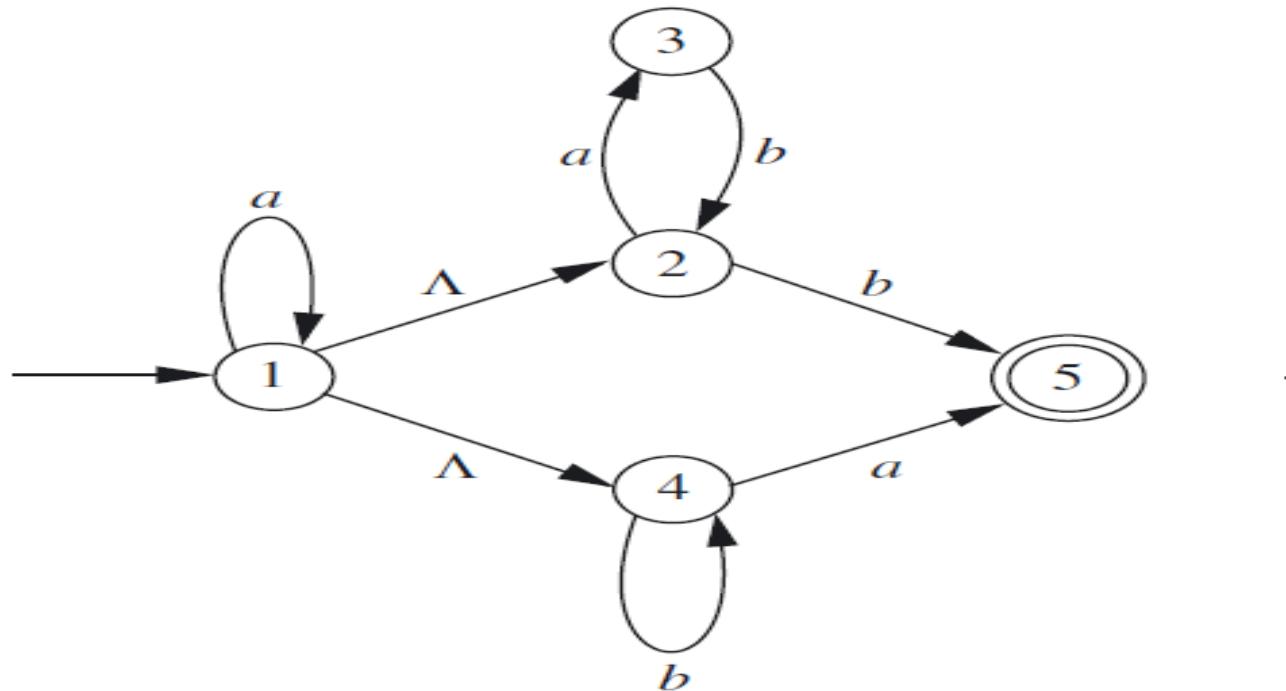
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2						
3						
4						
5						



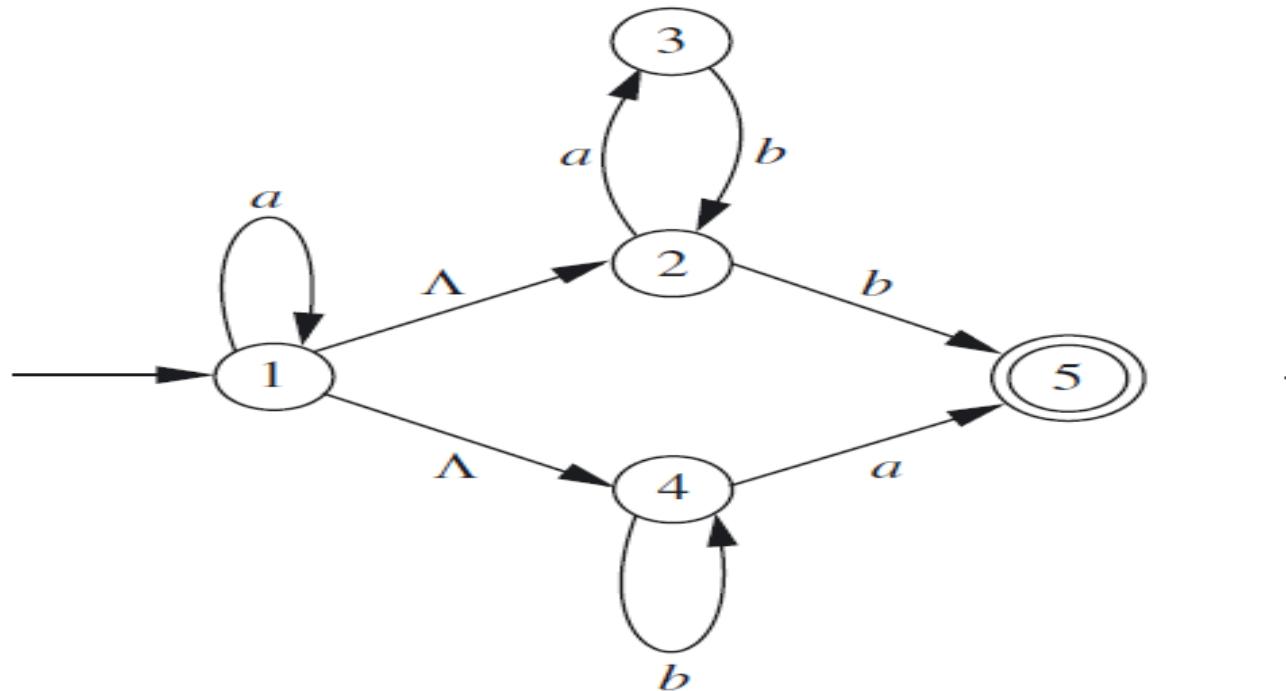
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$					
3						
4						
5						



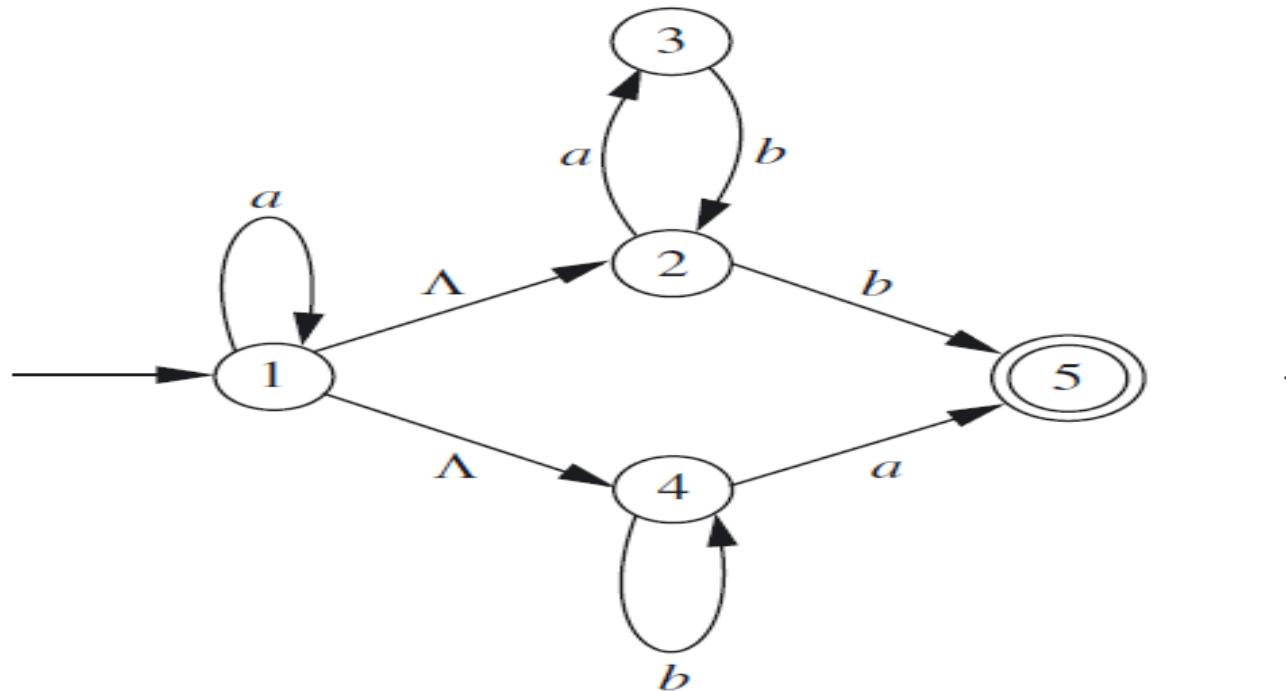
	$\lambda$	$a$			$b$	
	$\lambda$	$\lambda$	$\lambda$	$\lambda$	$\lambda$	$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$				
3						
4						
5						



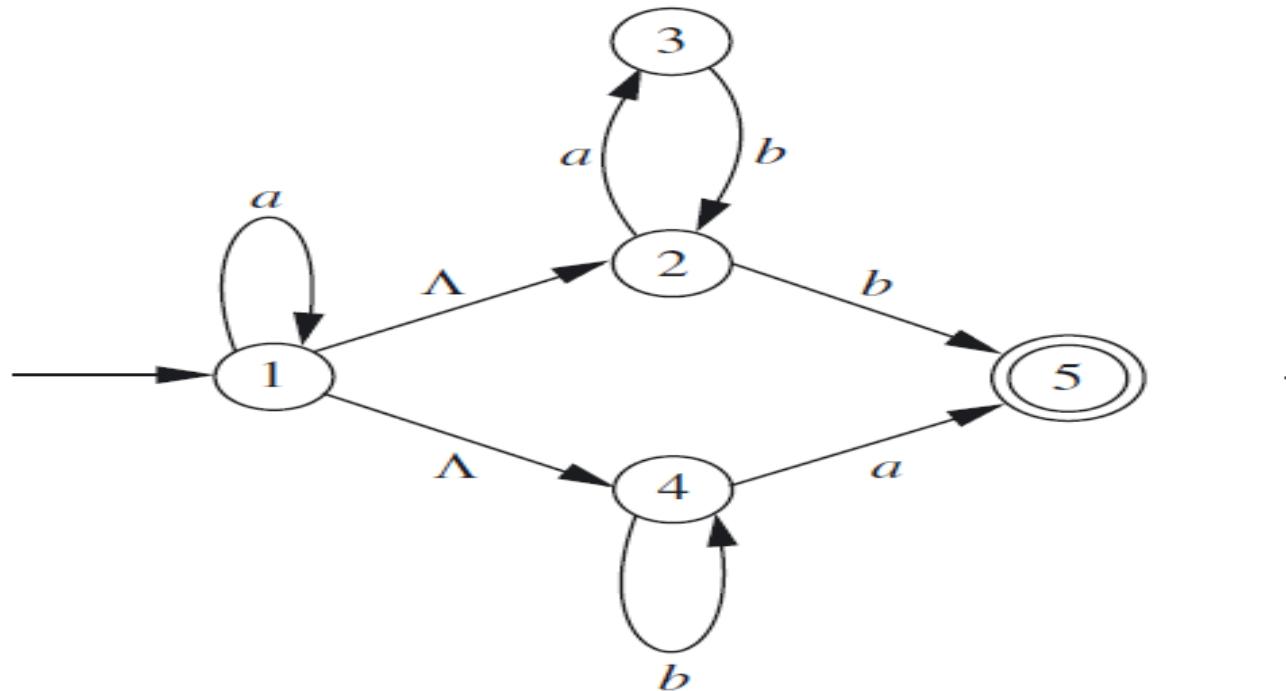
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$			
3						
4						
5						



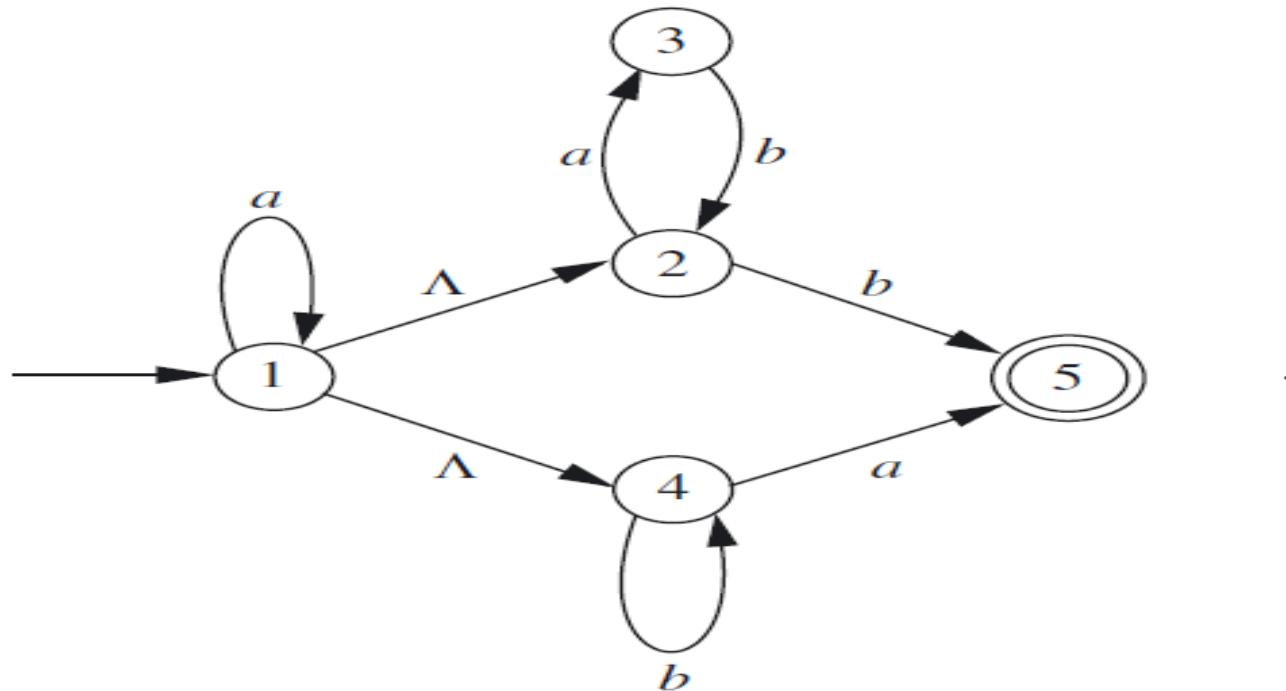
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$		
3						
4						
5						



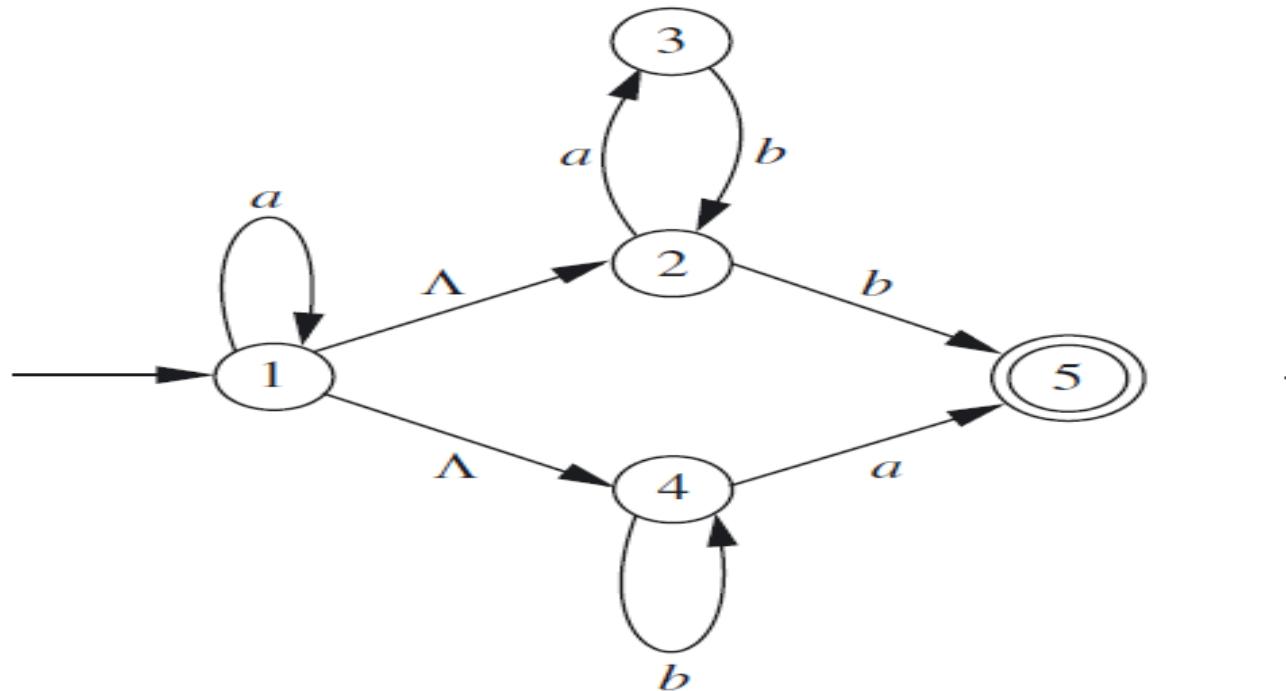
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	
3						
4						
5						



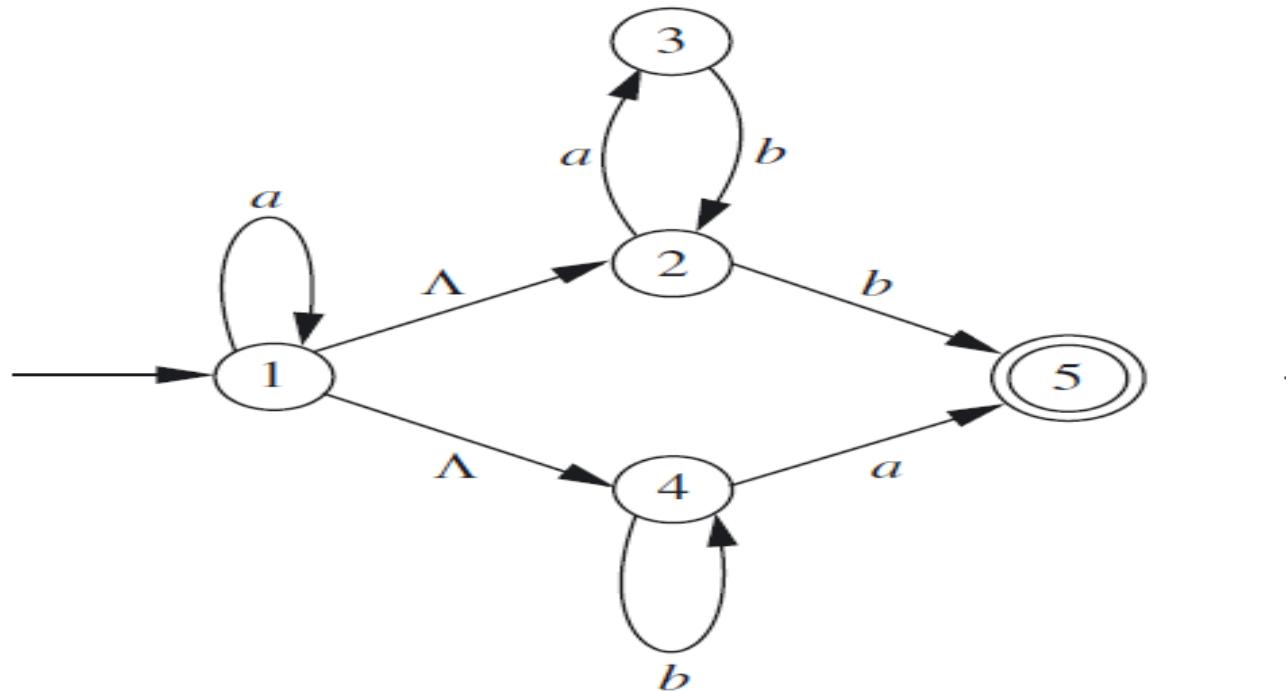
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3						
4						
5						



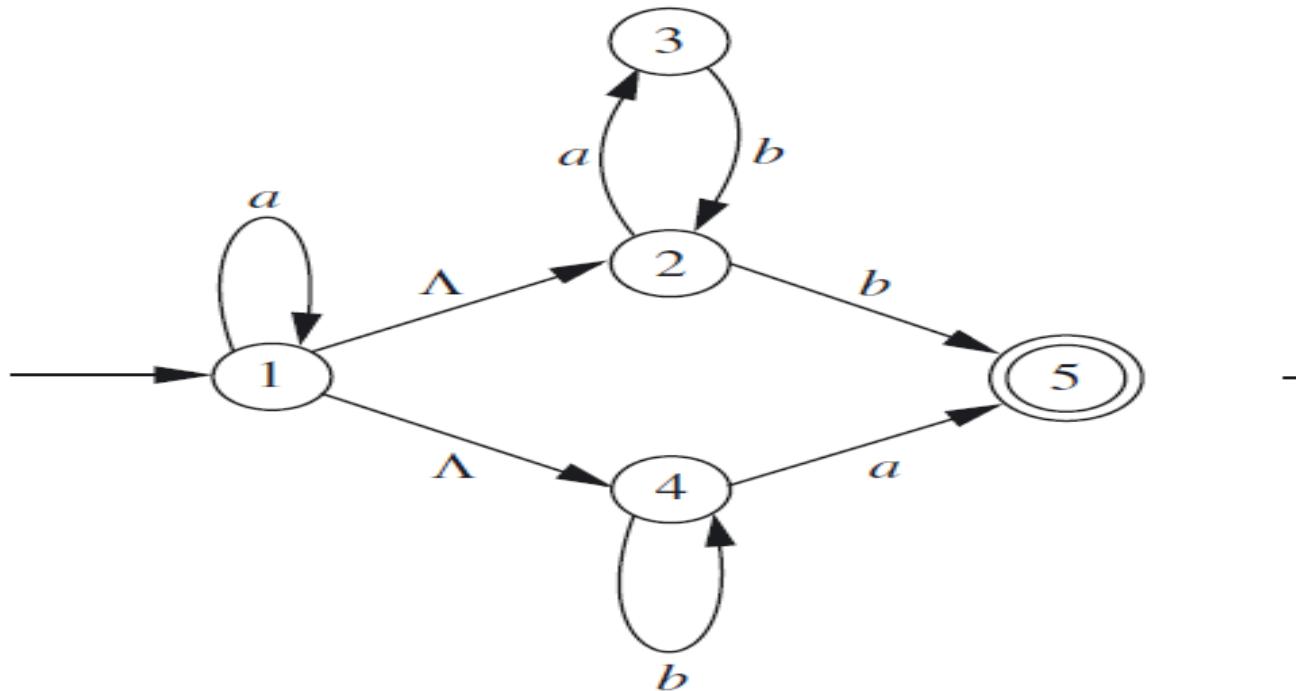
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$					
4						
5						



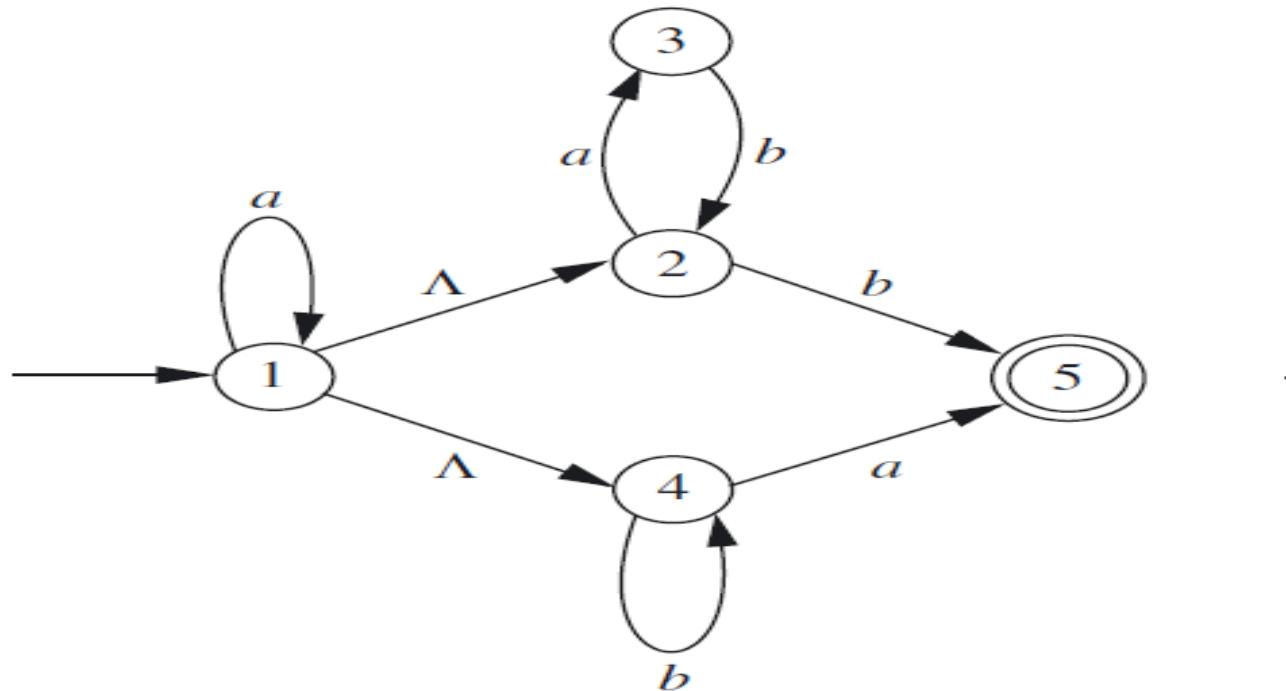
	$\lambda$	a			b	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$				
4						
5						



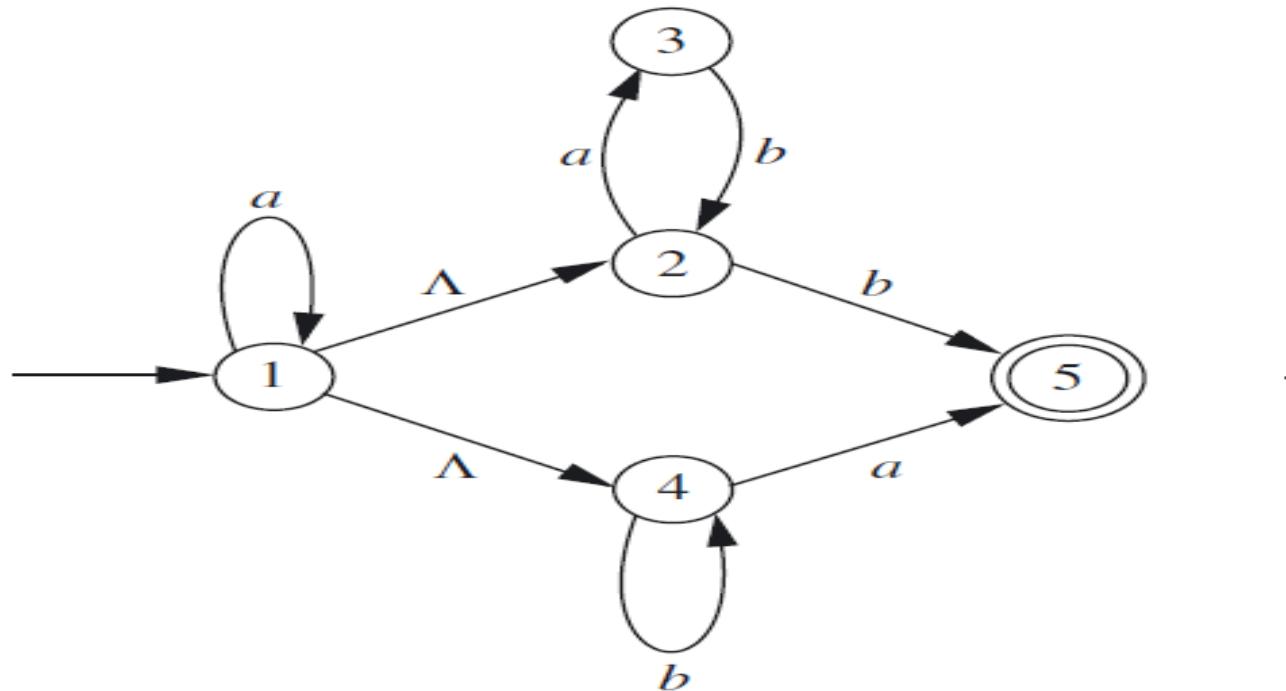
	$\lambda$	a			b	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$			
4						
5						



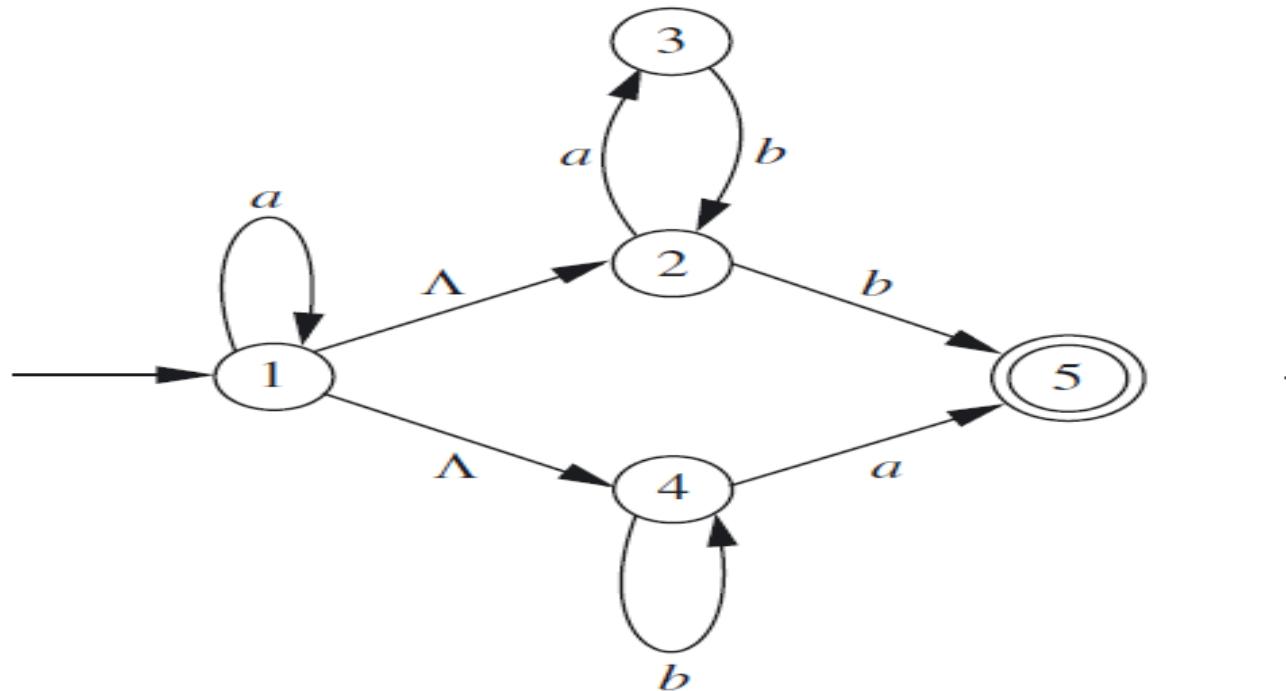
	$\lambda$	a			b	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$		
4						
5						



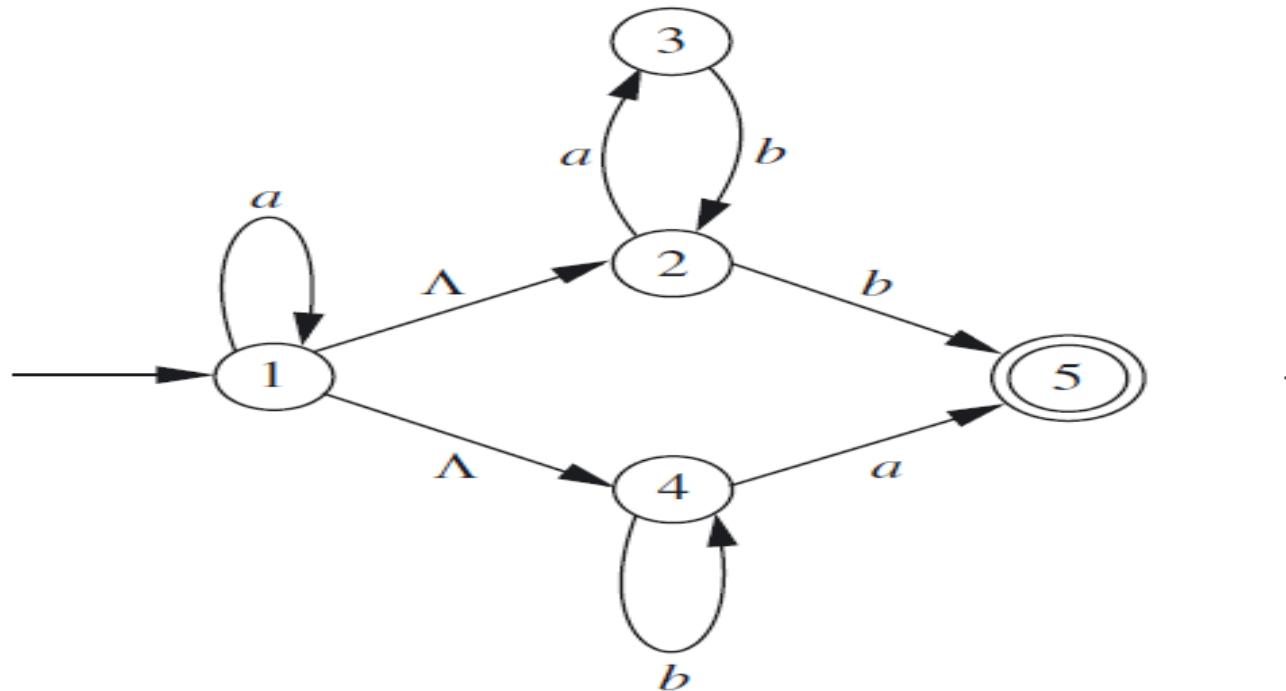
	$\lambda$	$a$			$b$	
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	
4						
5						



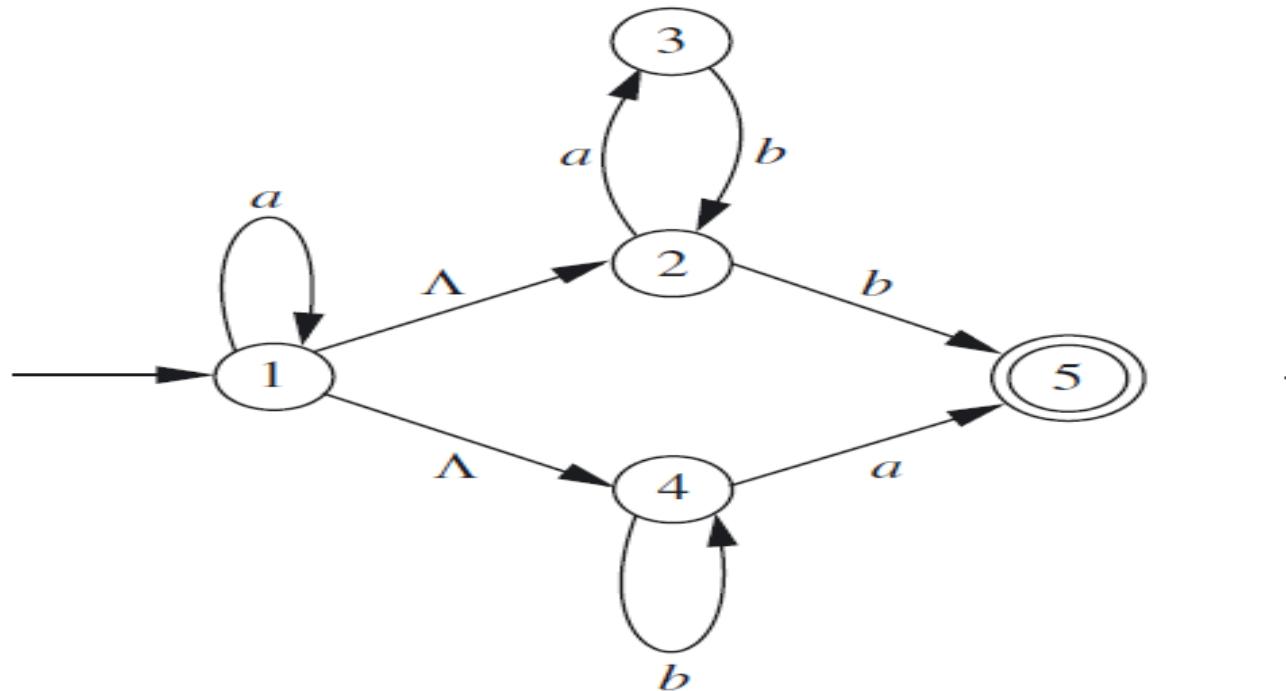
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4						
5						



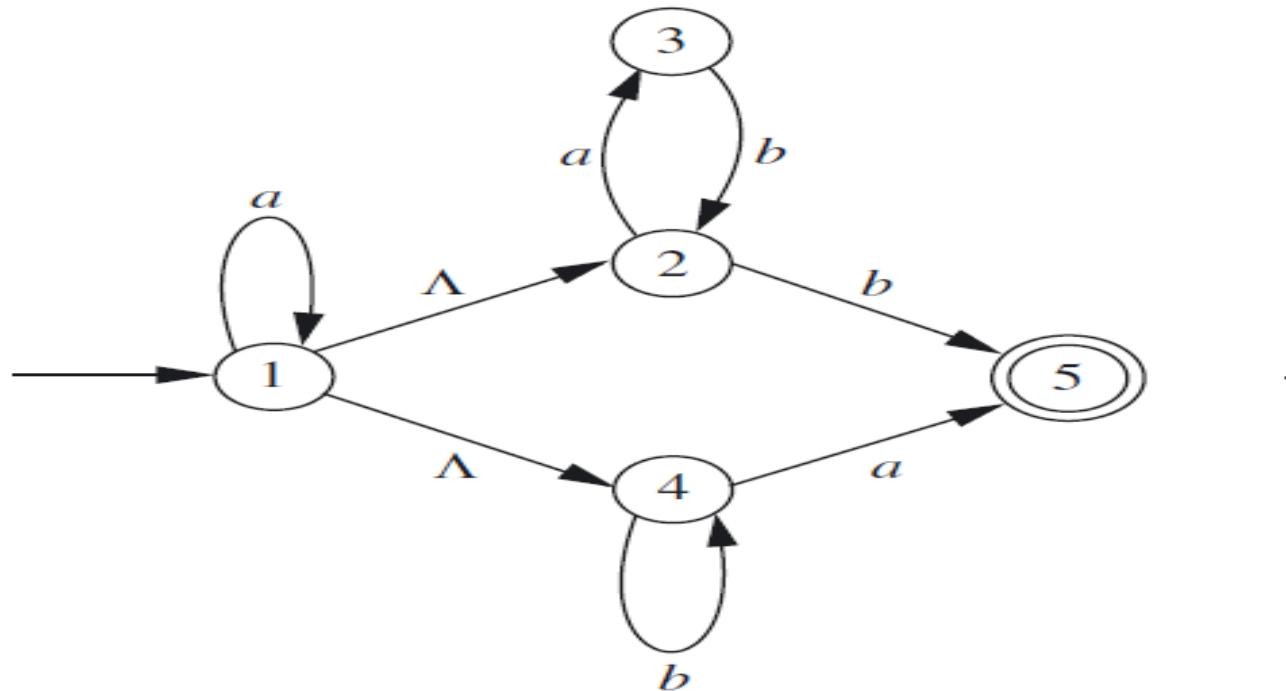
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$					
5						



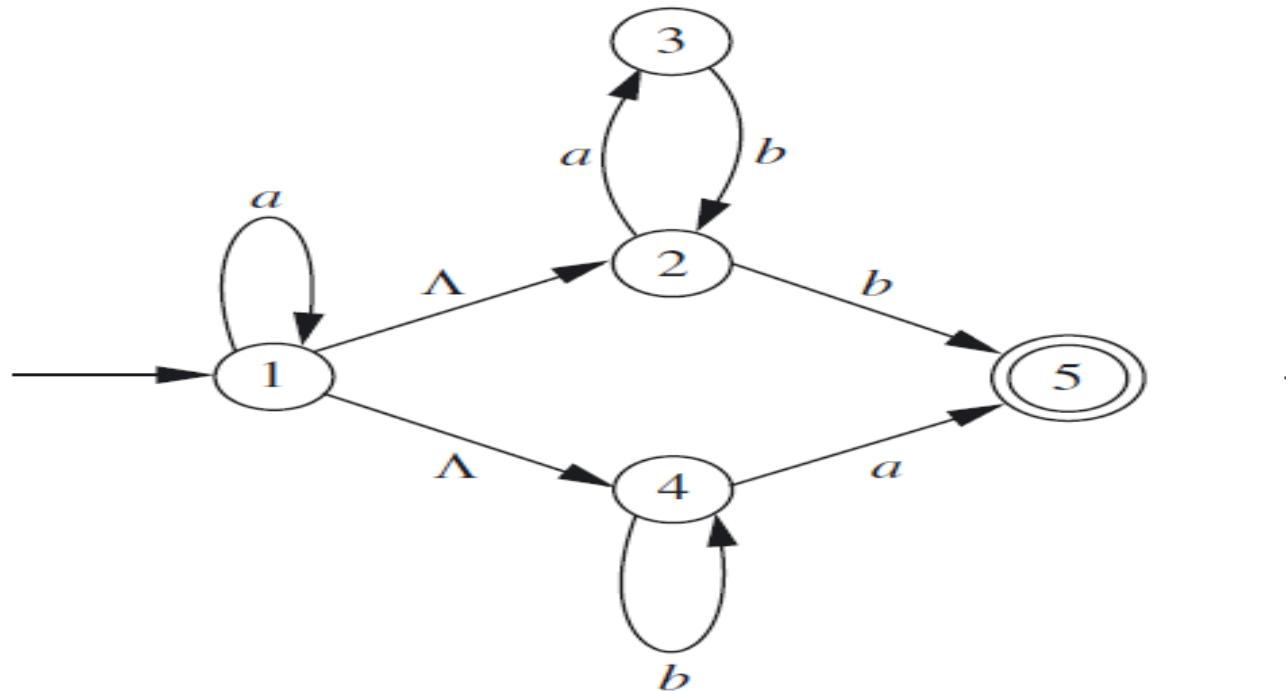
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$				
5						



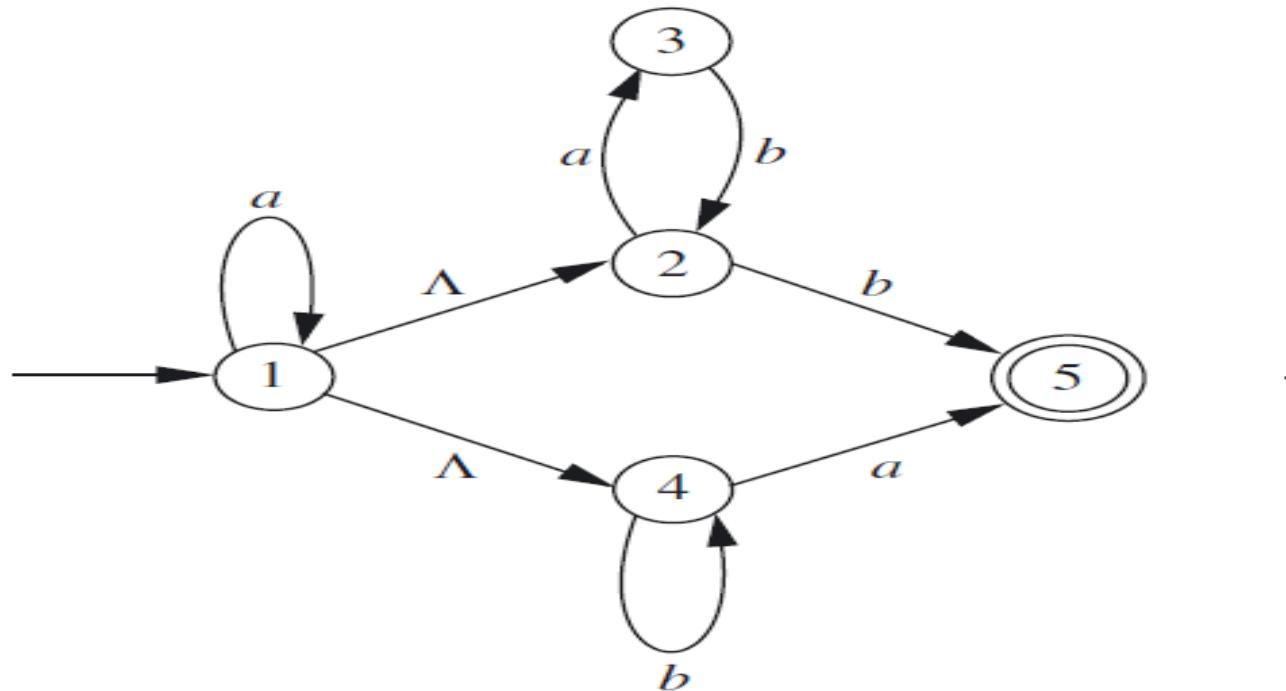
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$			
5						



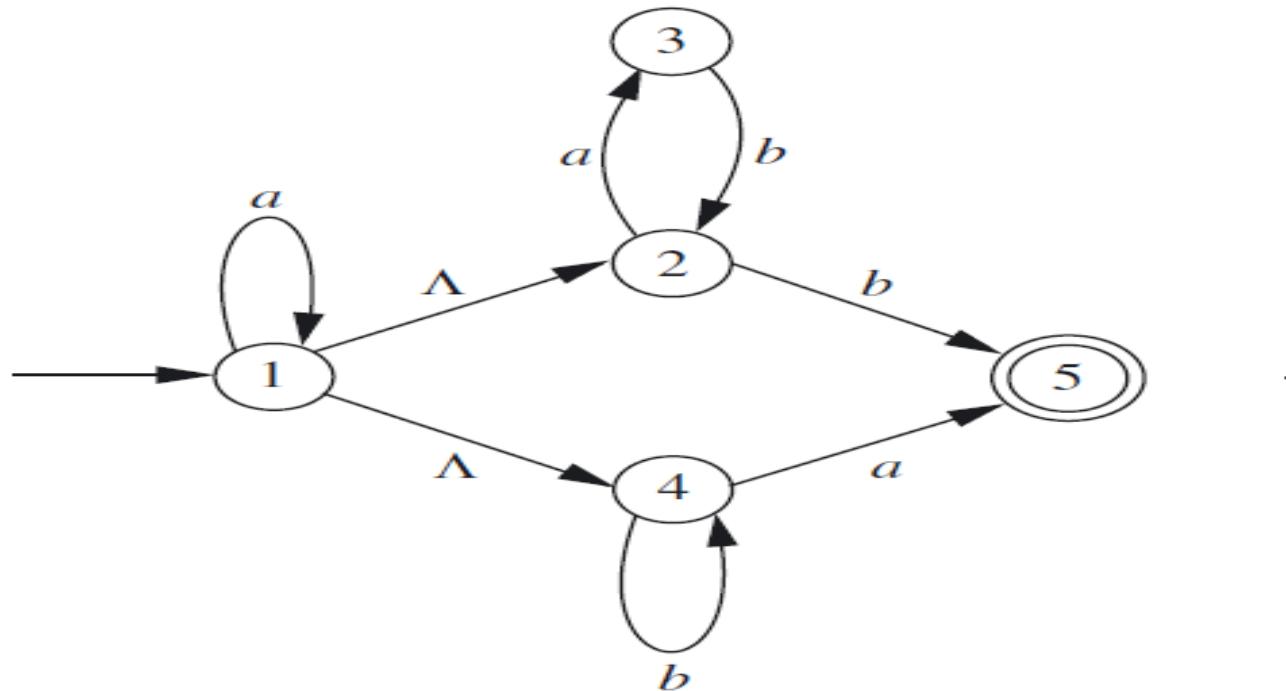
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$		
5						



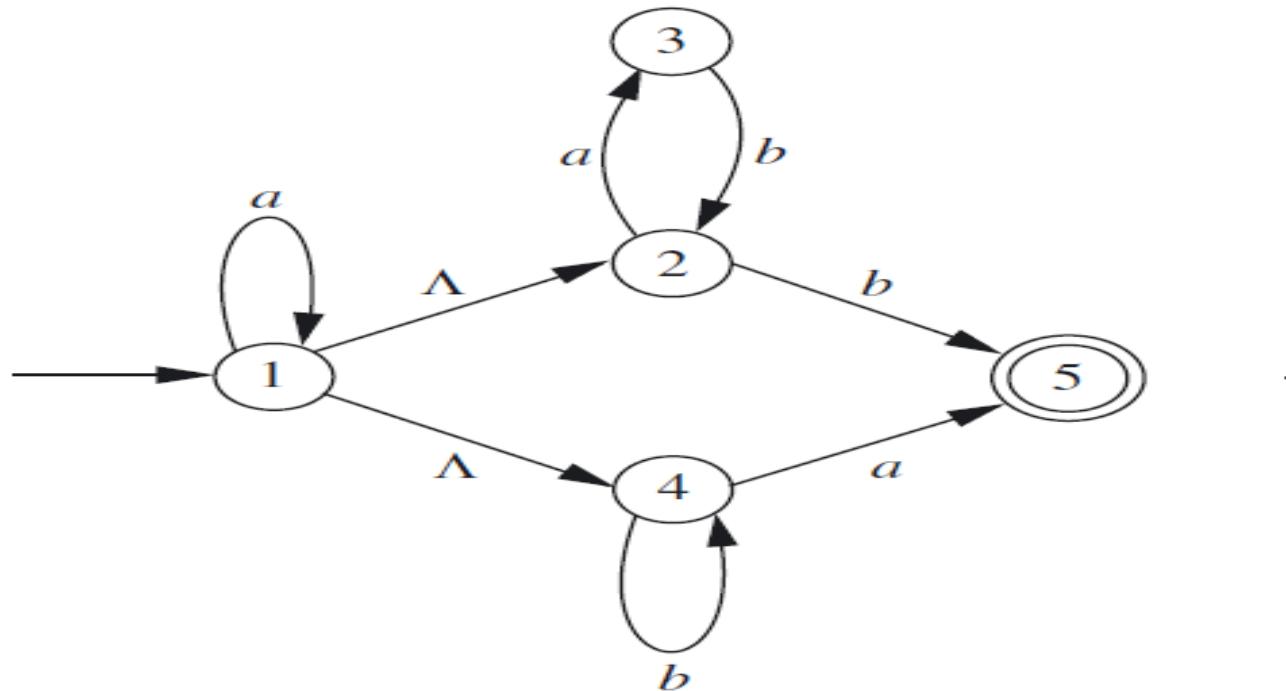
		a			b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	
5						



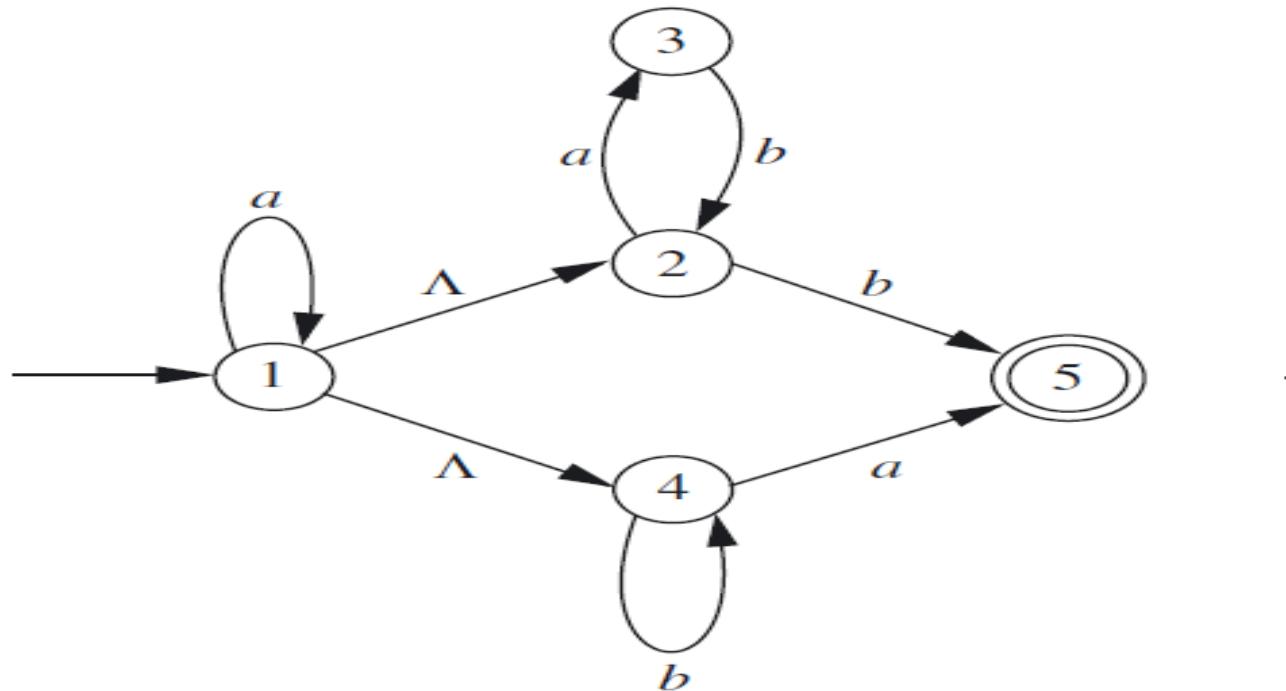
		a			b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5						



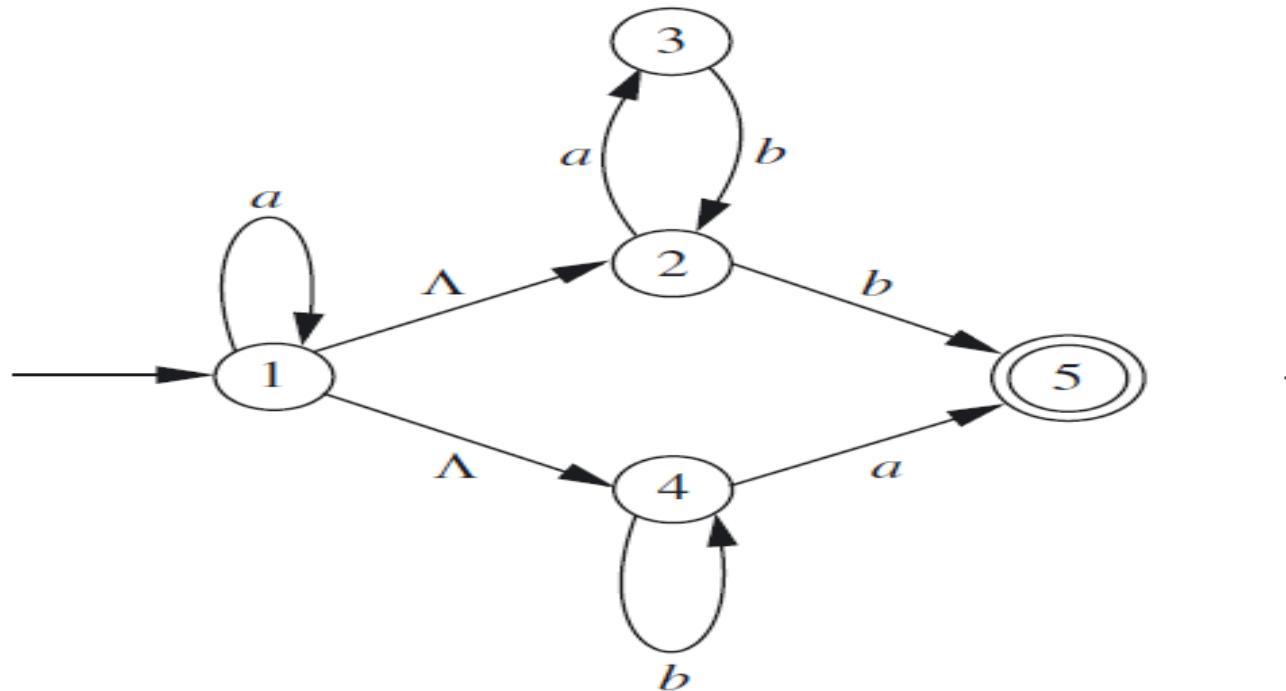
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$					



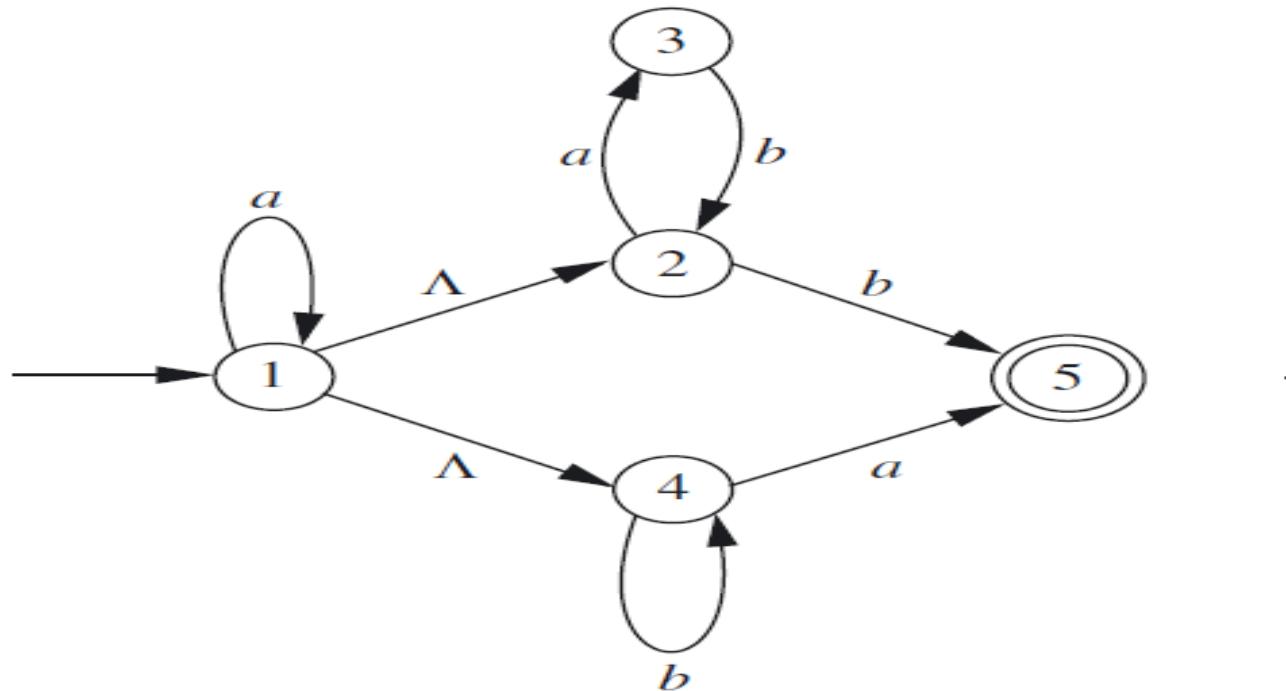
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$				



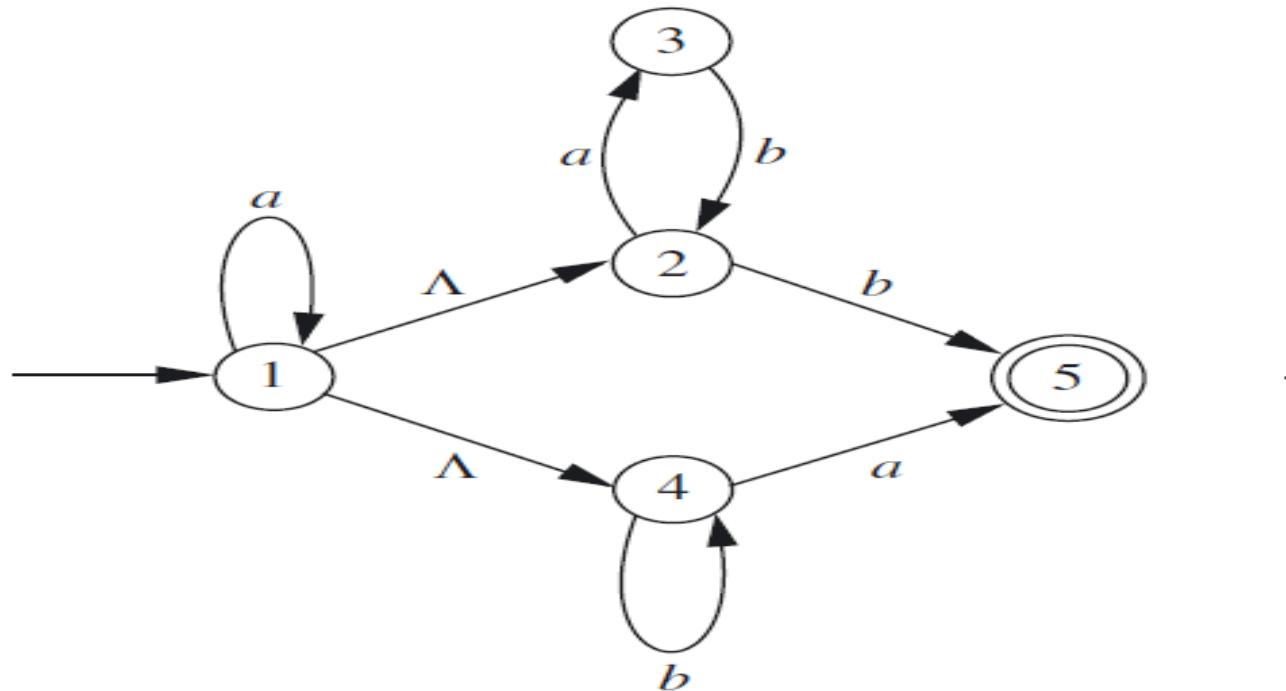
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$			



		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$		

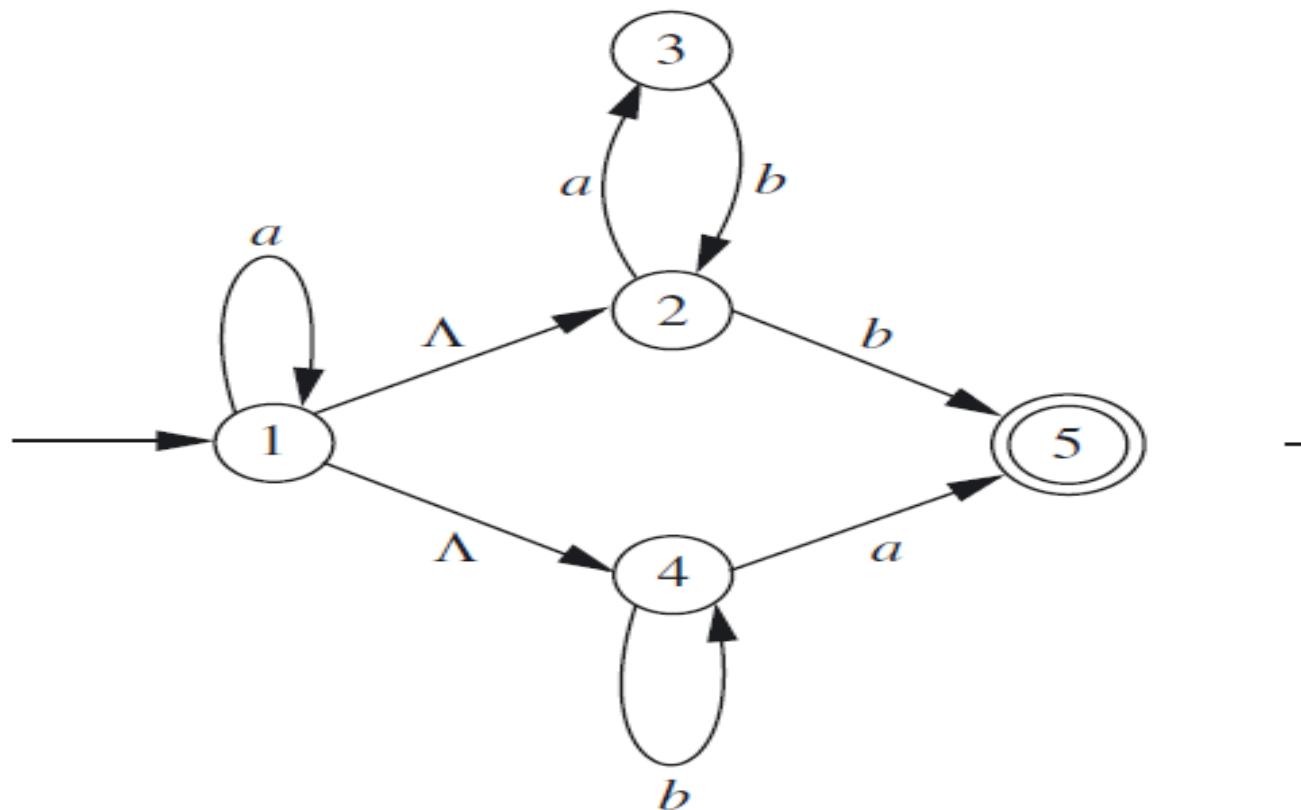


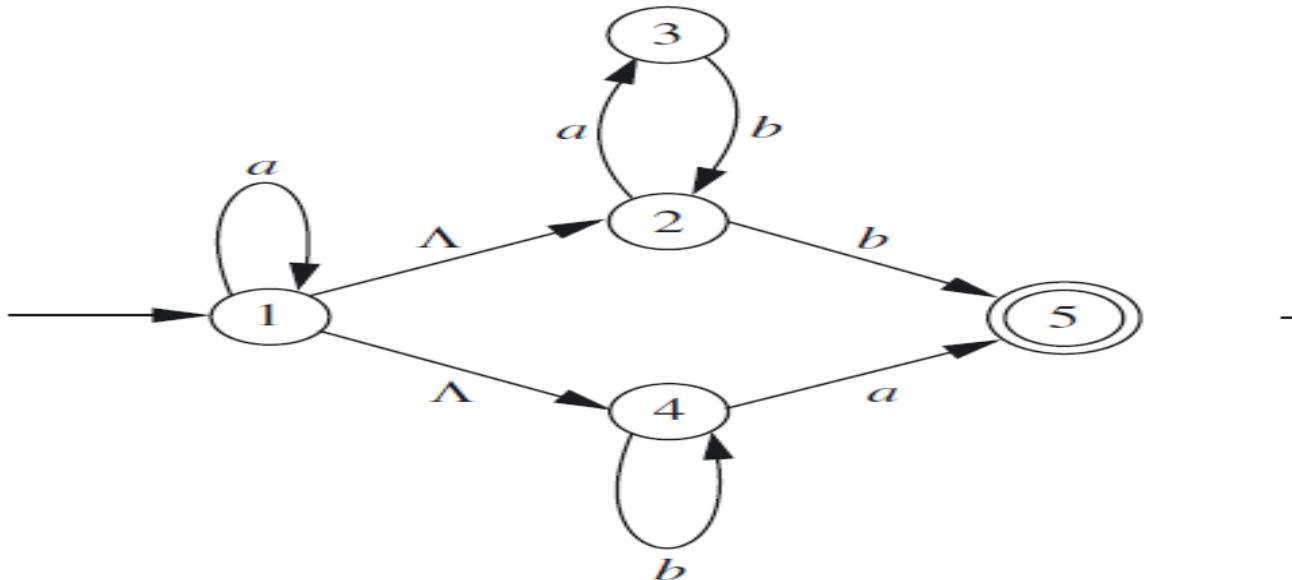
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	

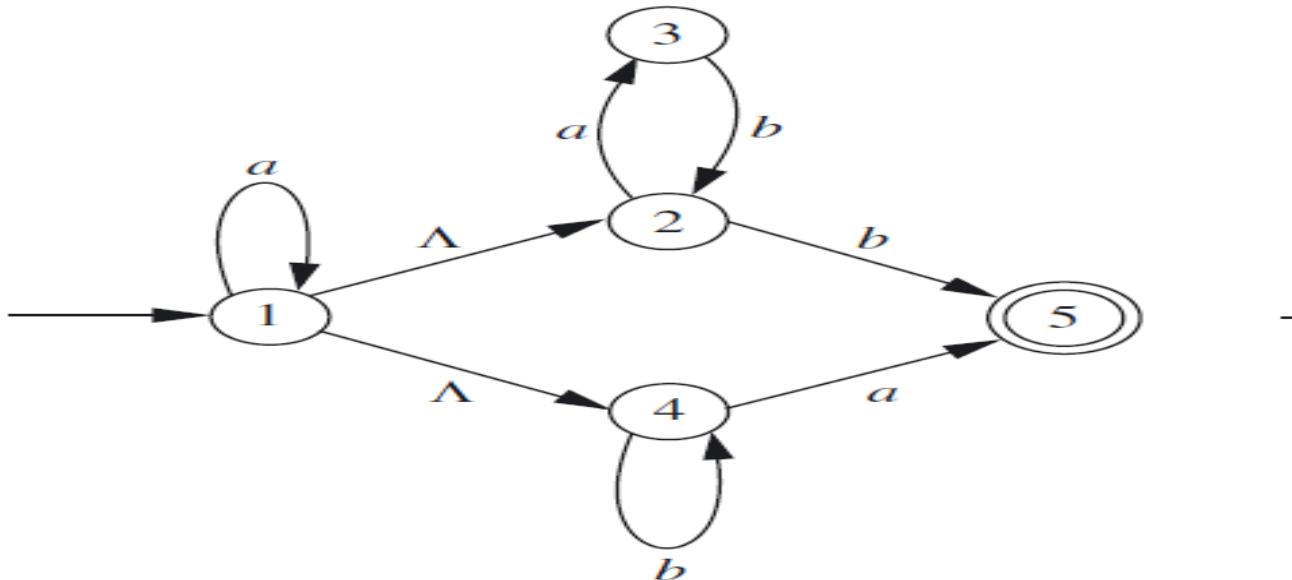


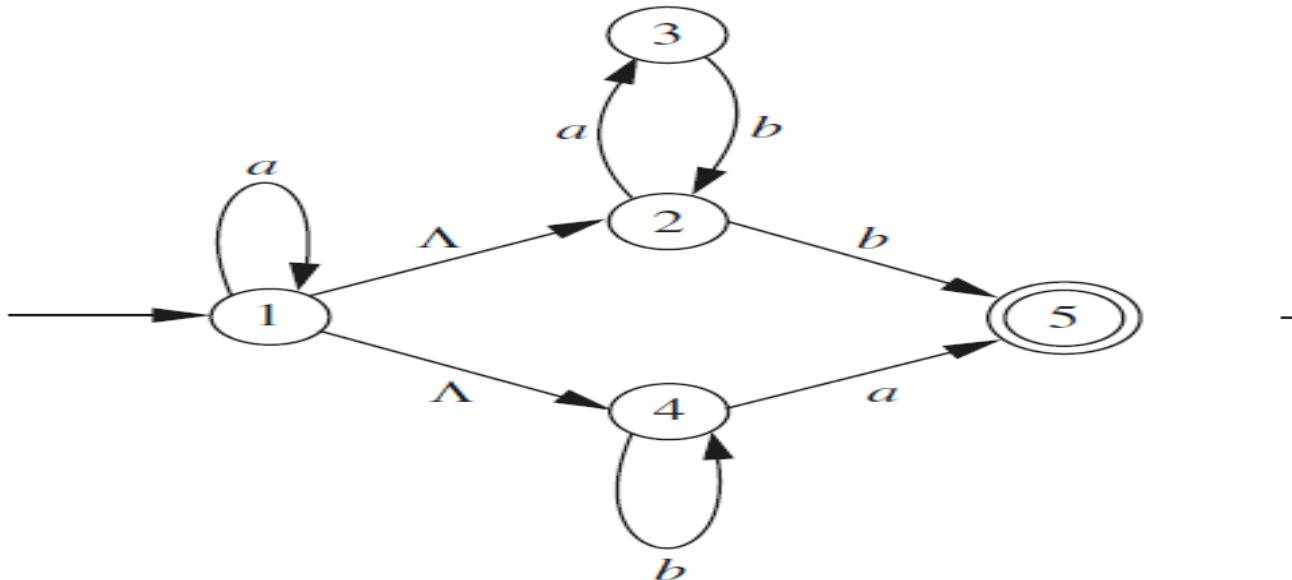
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
3	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$

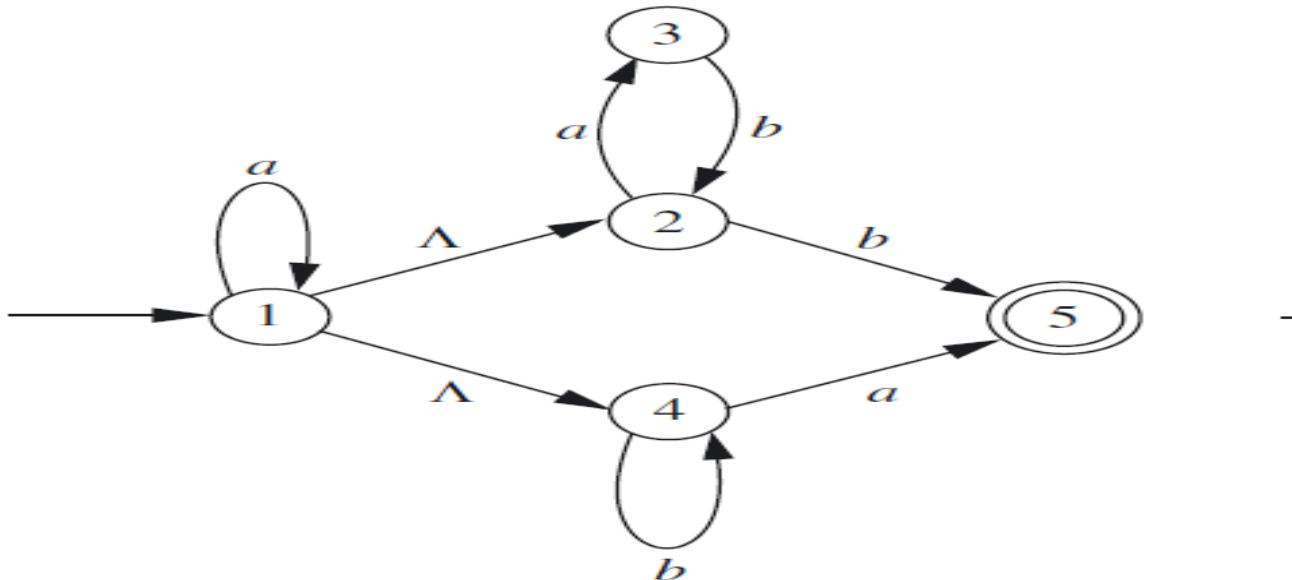
# Example for NFA-null to DFA Conversion

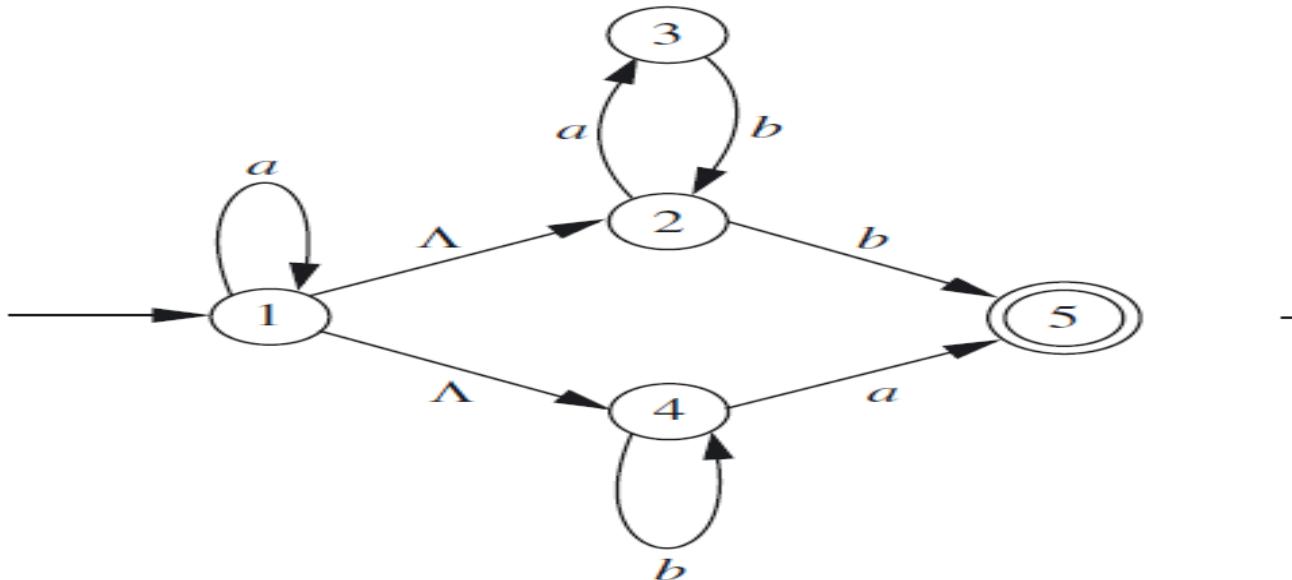


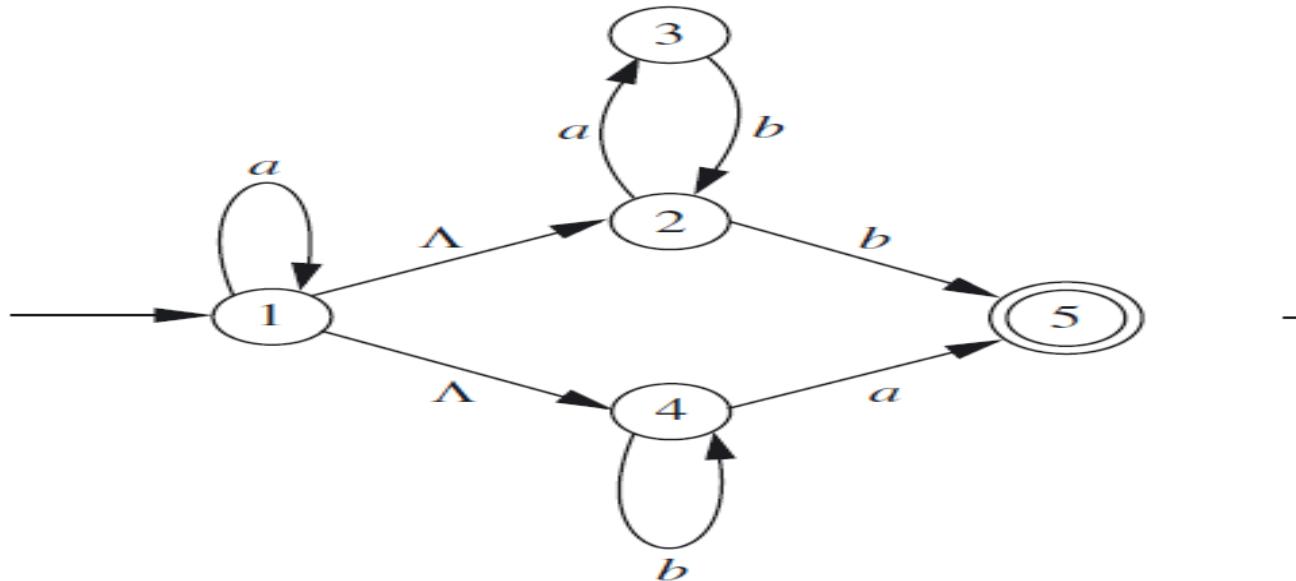




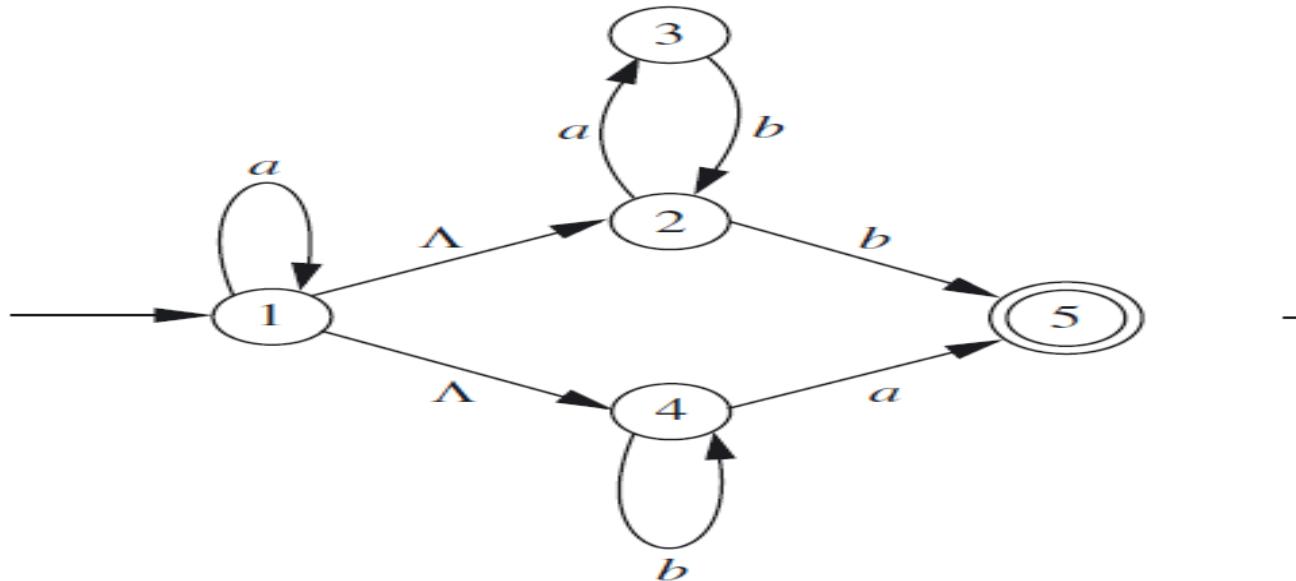




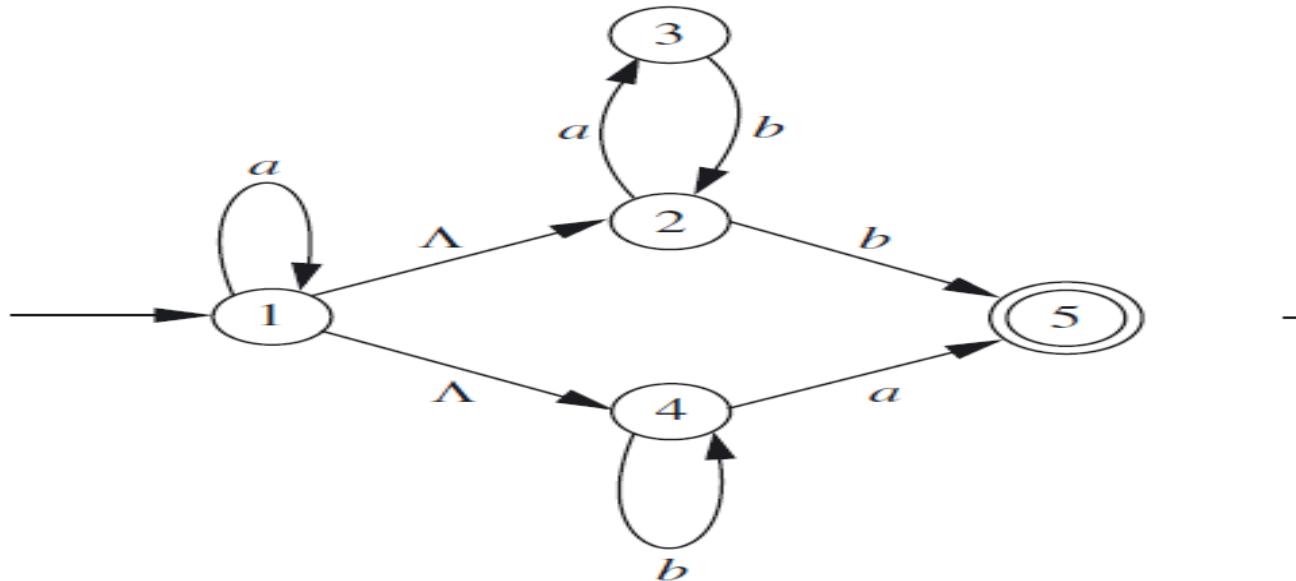




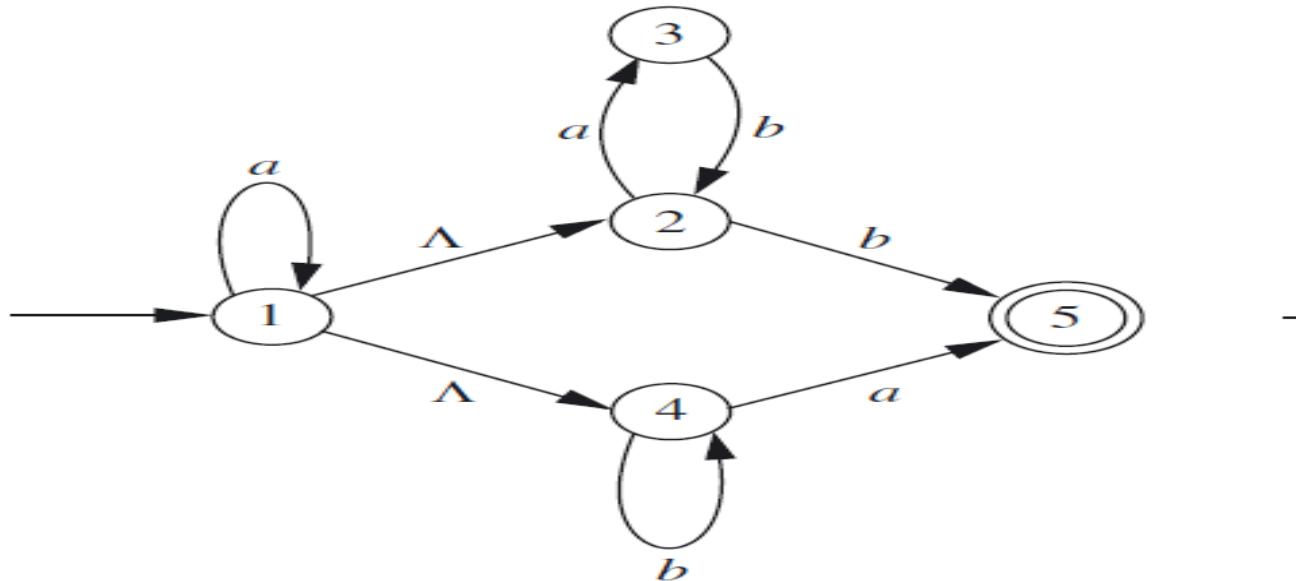
		$a$			$b$	
1	$\lambda$	$\{1,3,5\}$	$\lambda$	$\{1,2,3,4,5\}$	$\lambda$	$\{4,5\}$



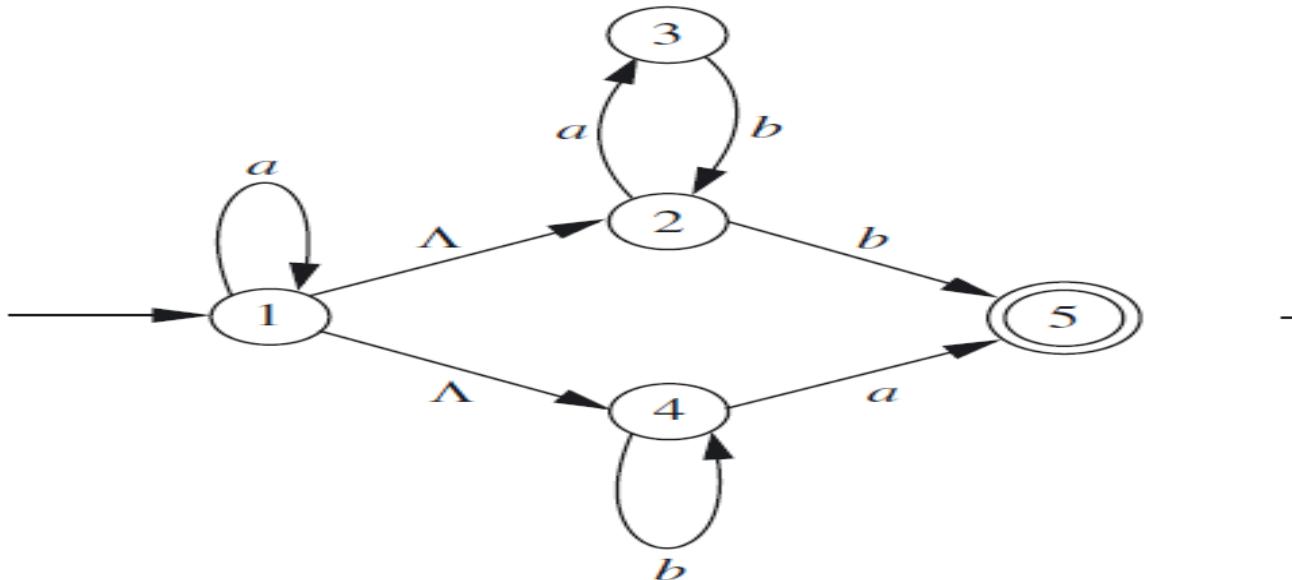
		$a$			$b$	
1	$\lambda$	$\{1,3,5\}$	$\lambda$	$\{1,2,3,4,5\}$	$\lambda$	$\{4,5\}$

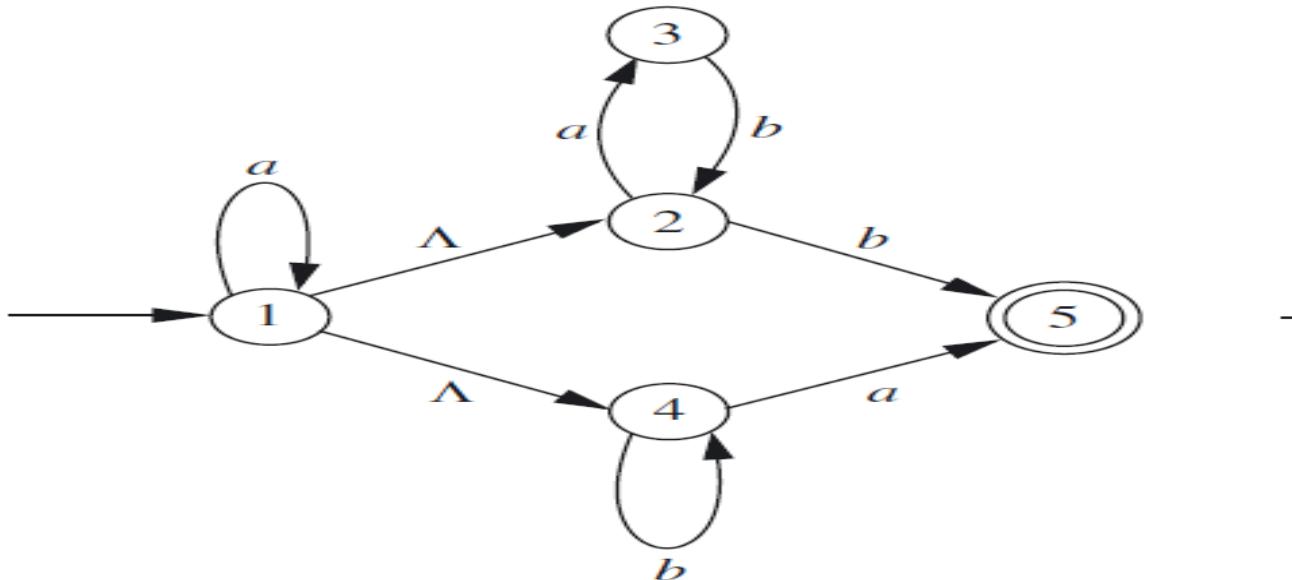


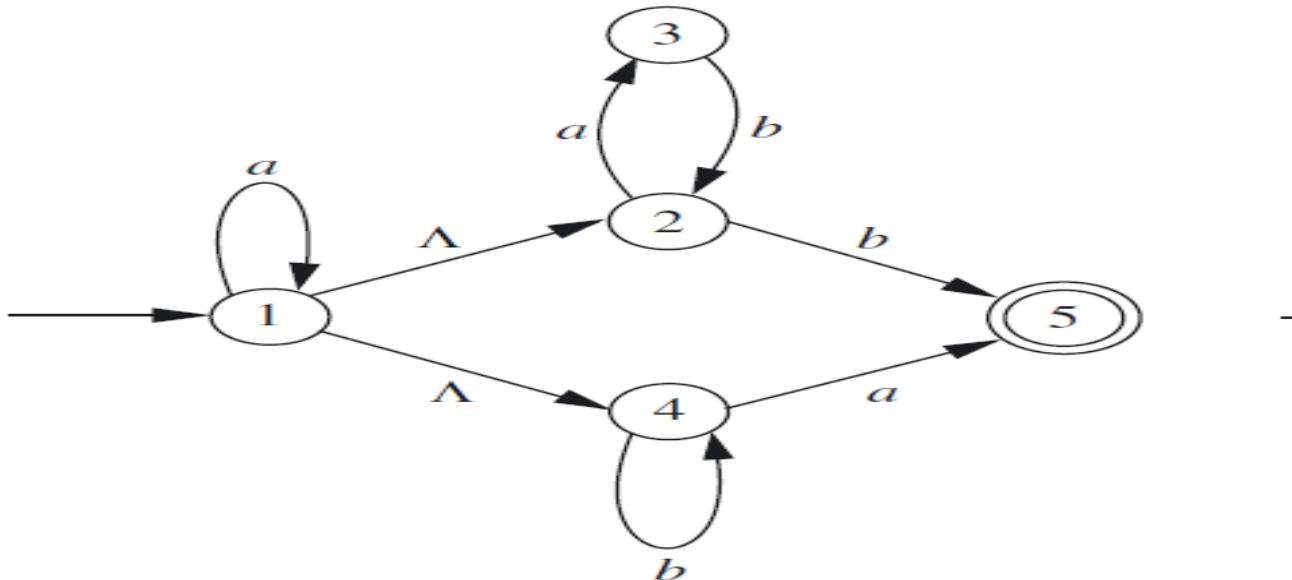
		$a$			$b$	
1	$\lambda$	$\{1,3,5\}$	$\lambda$	$\{1,2,3,4,5\}$	$\lambda$	$\{4,5\}$

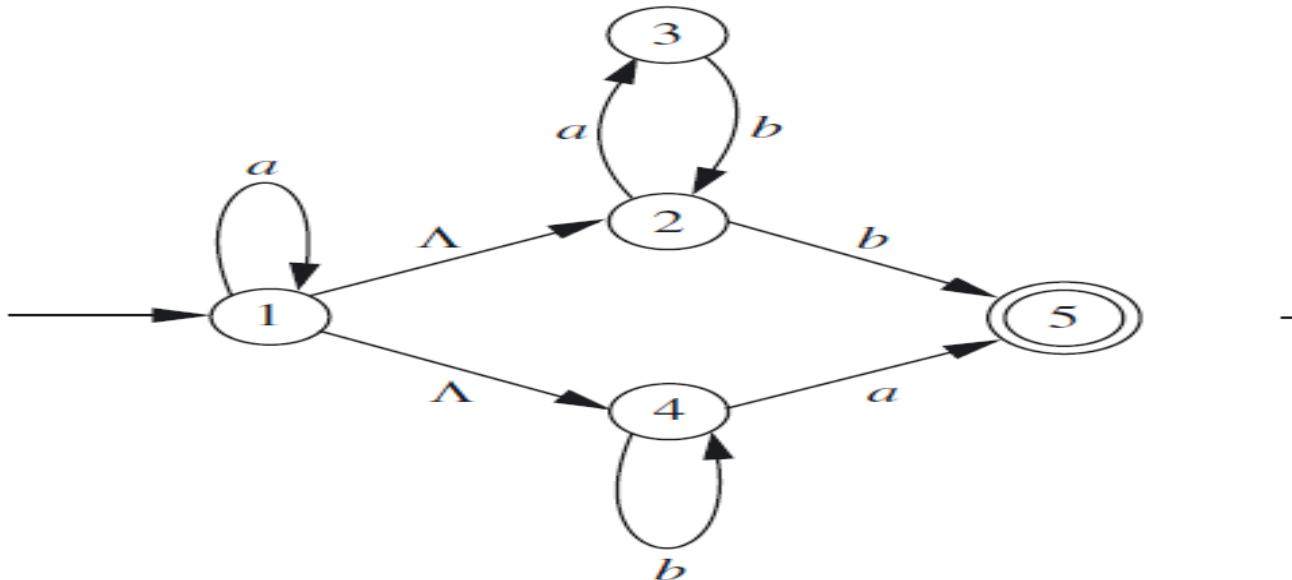


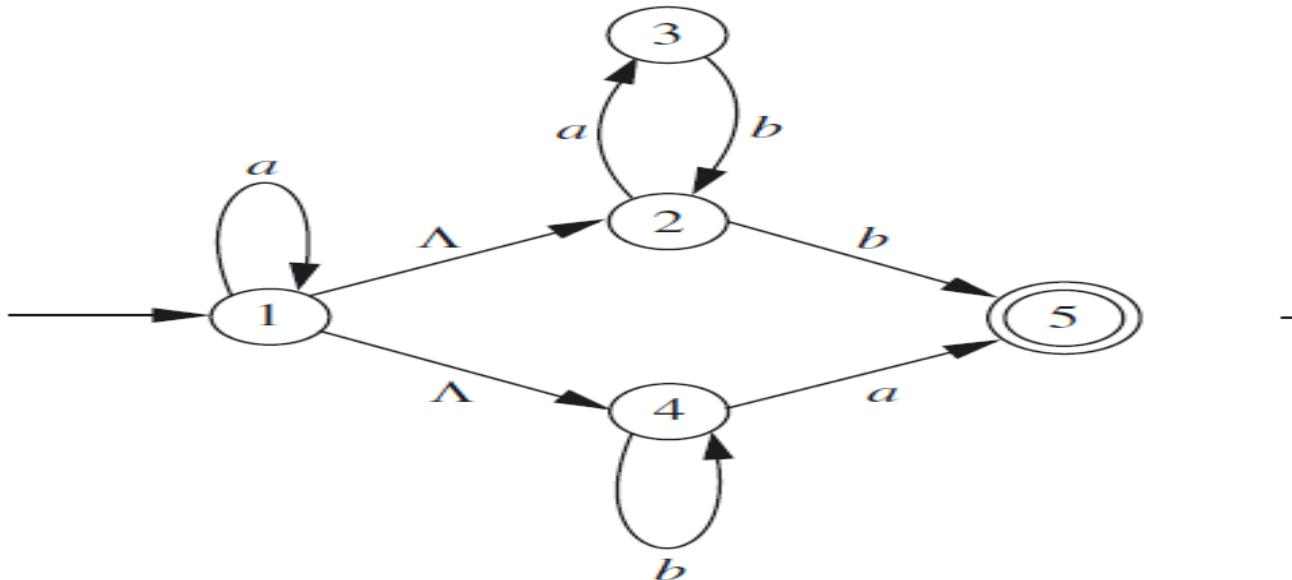
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$		$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$

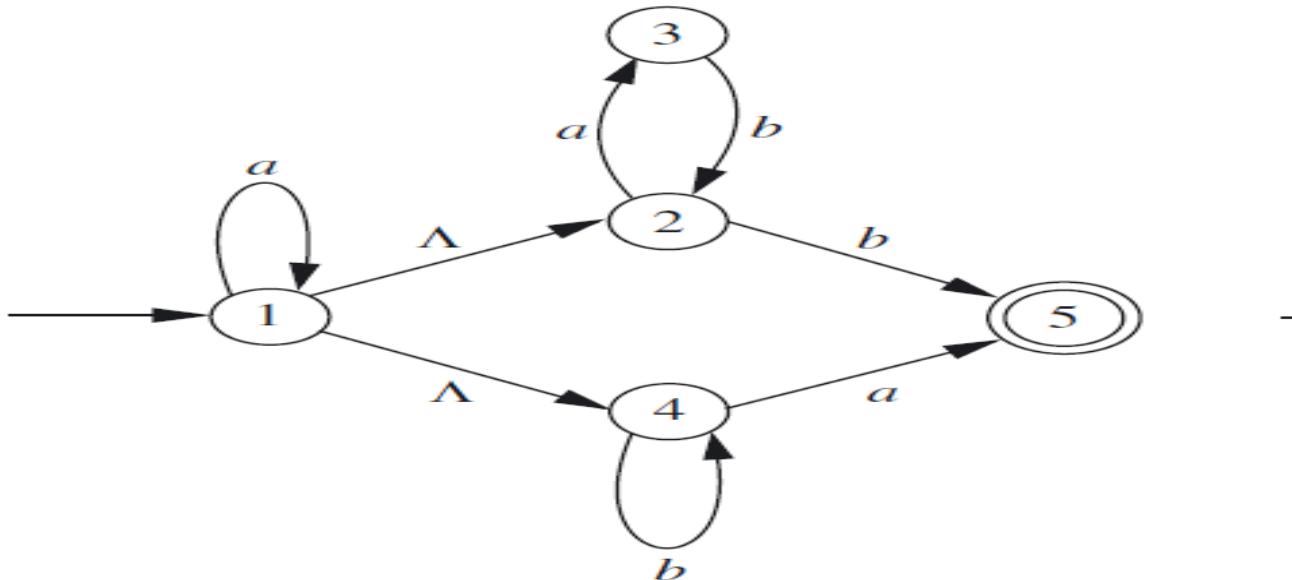




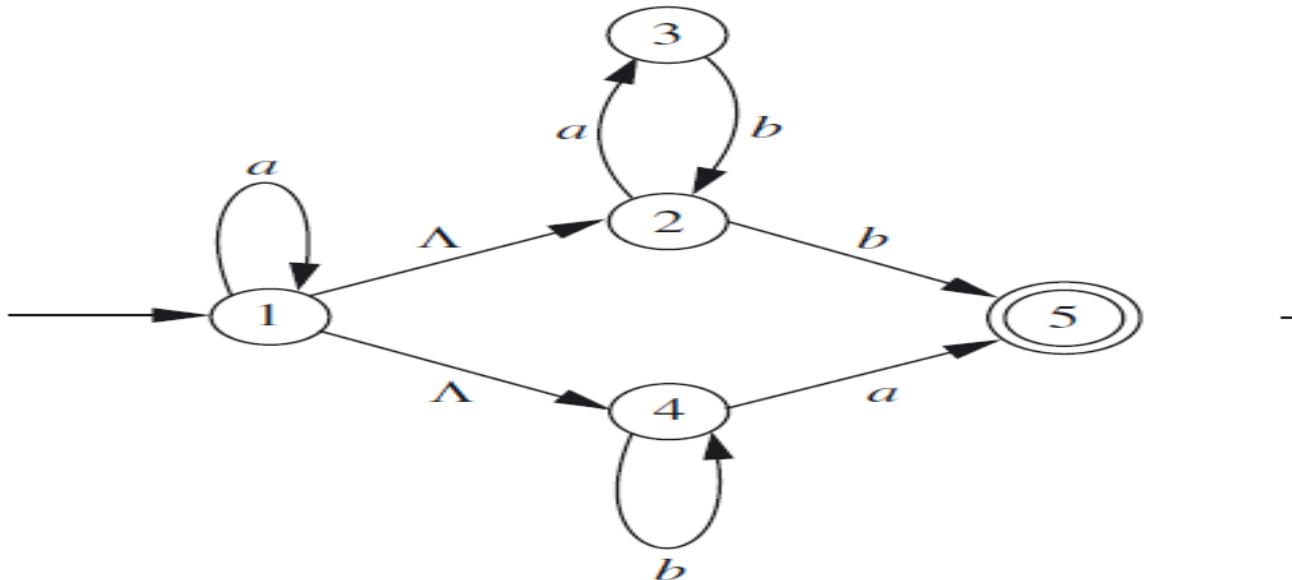


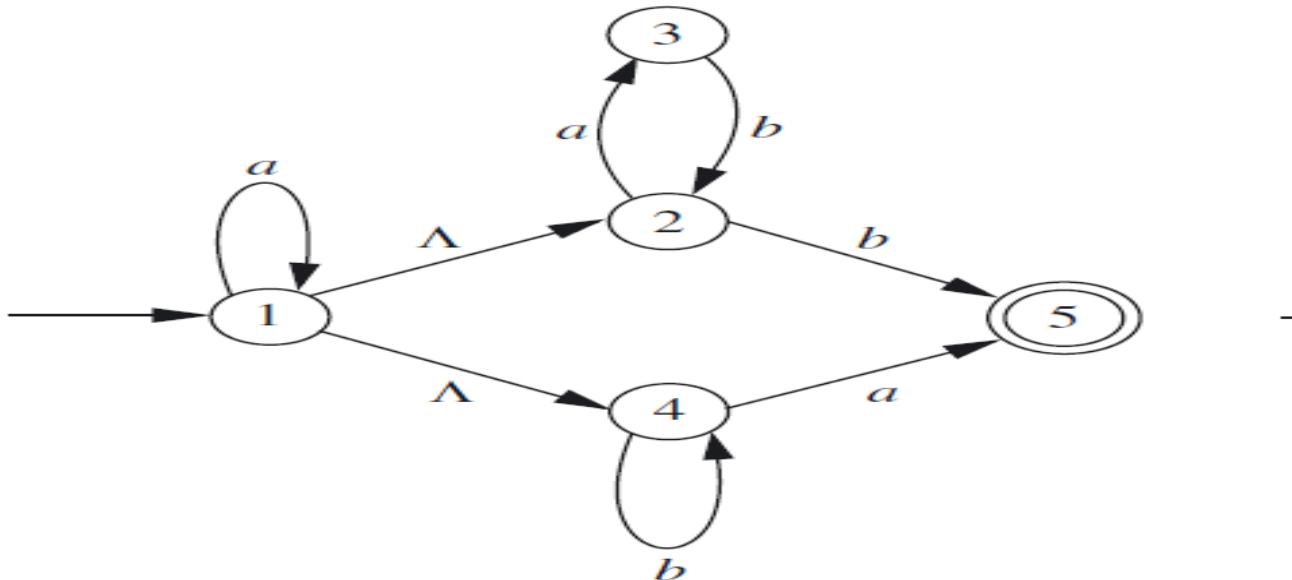


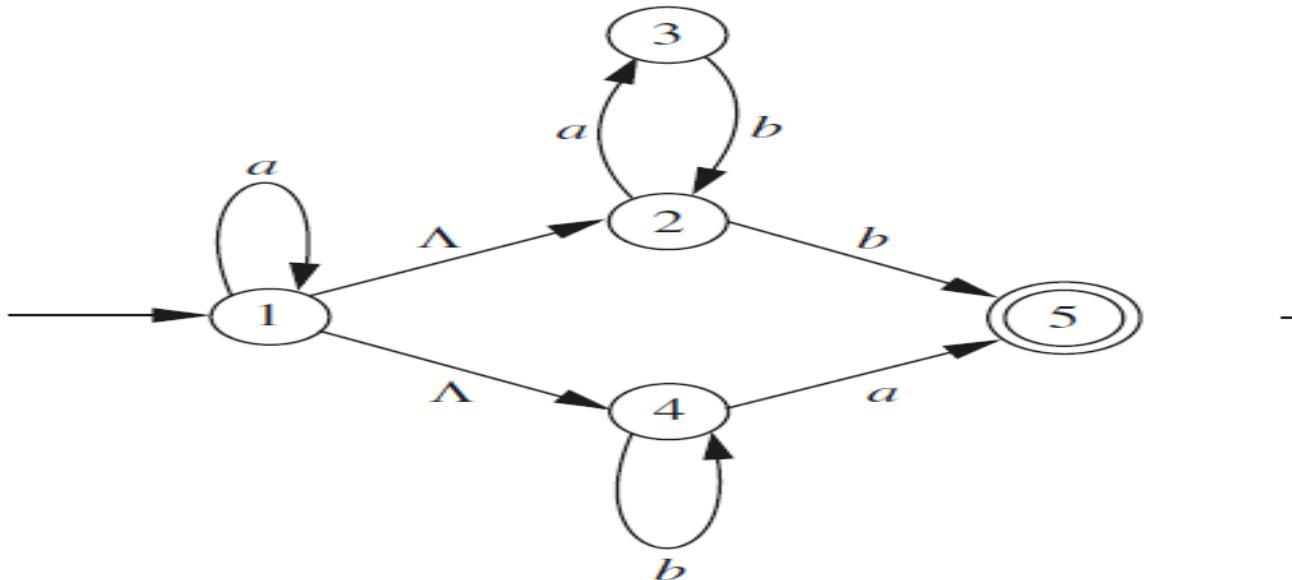


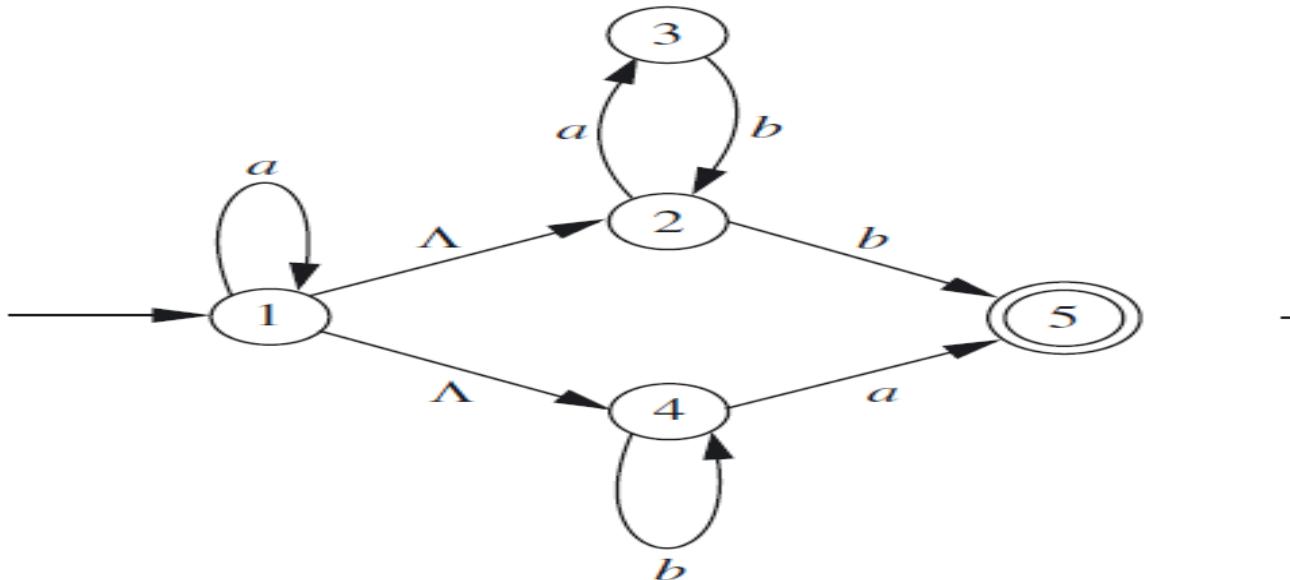


		a			b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$

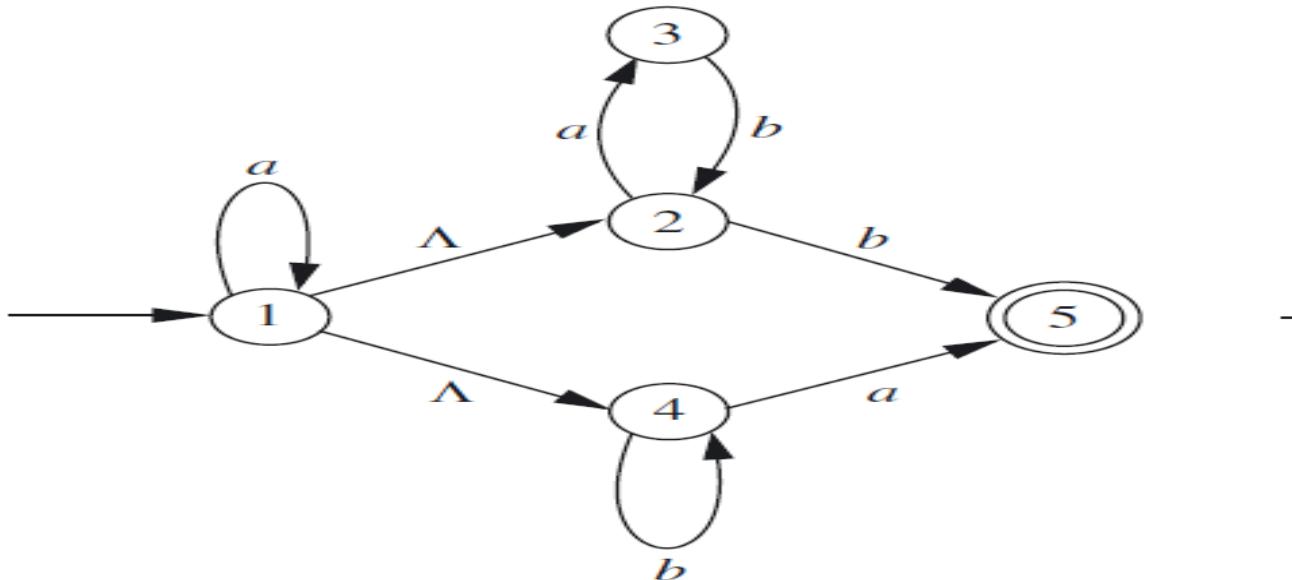




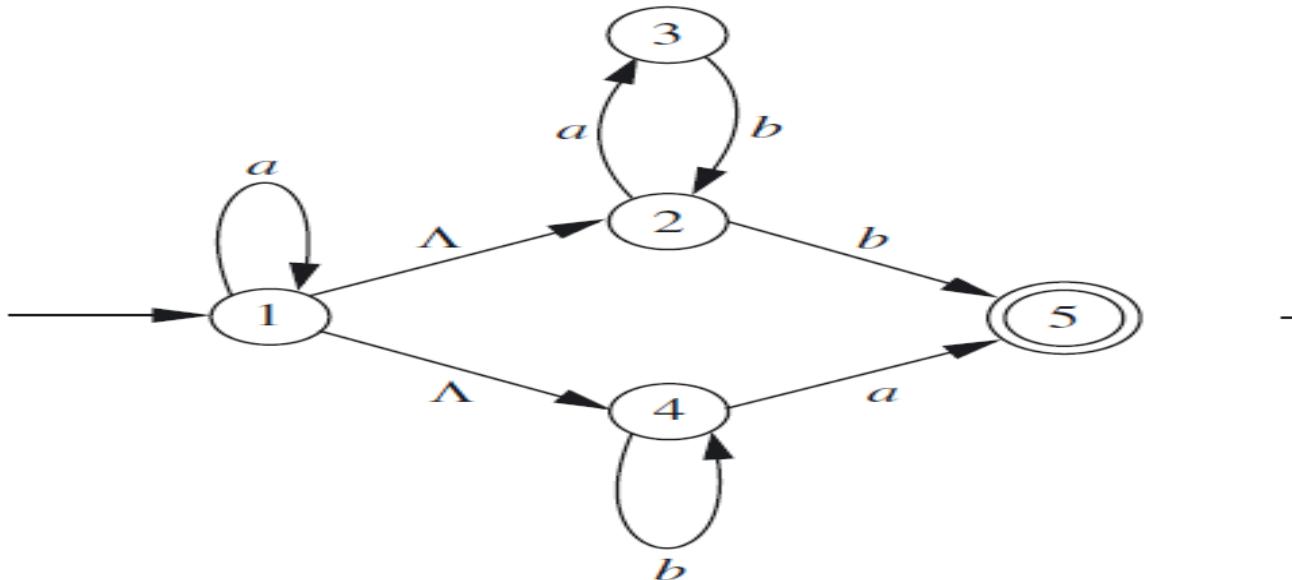




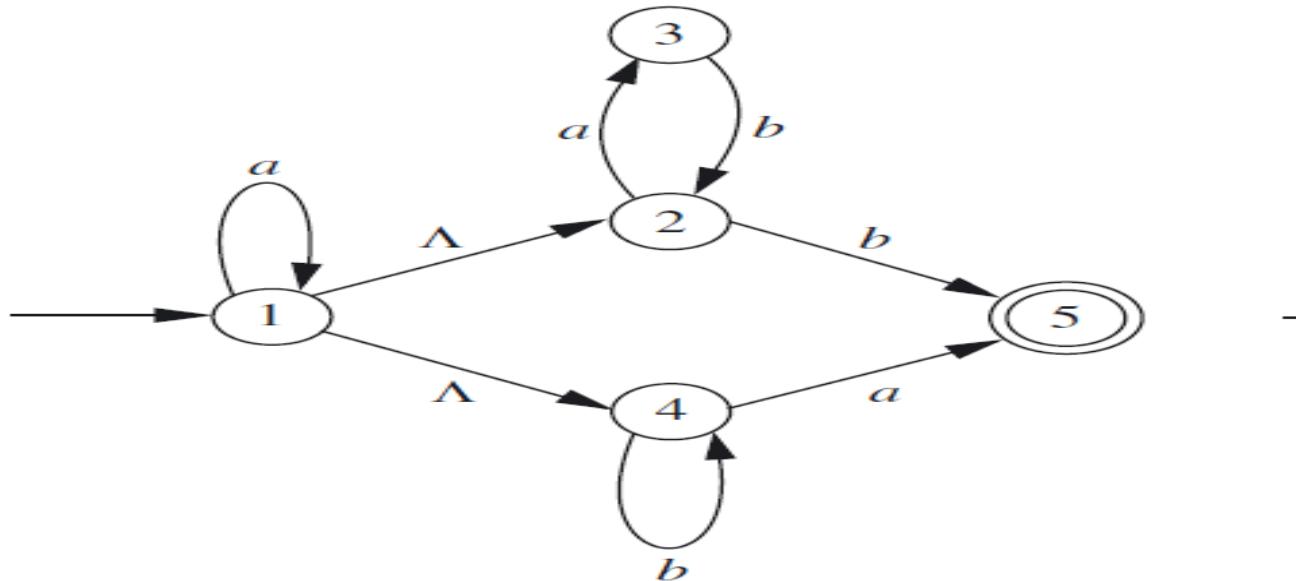




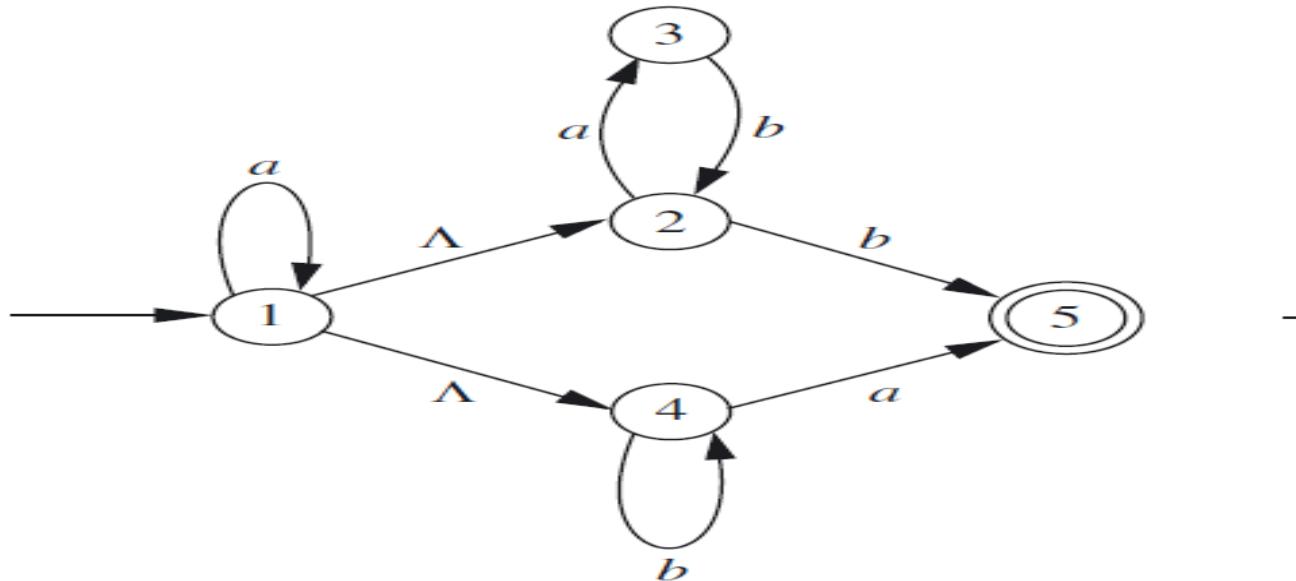
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	{1,2,4},	{1,3,5}	{1,2,3,4,5}	{1,2,4}	{4,5}	{4,5}
{1,2,3,4,5}	{1,2,3,4,5}	{1,2,3,4,5}	{1,2,3,4,5}	{1,2,3,4,5}	2,4,5	{2,4,5}
{4,5}	{4,5}	{5}	{5}	{4,5}	{4}	{4}



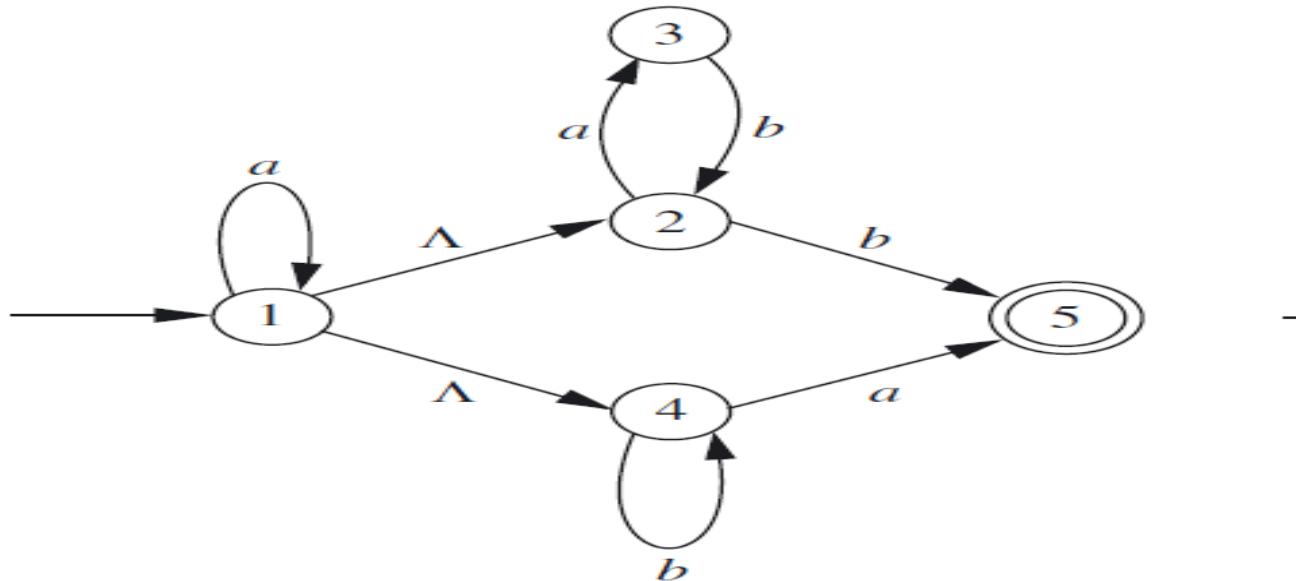
		a			b	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$2,4,5$	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$



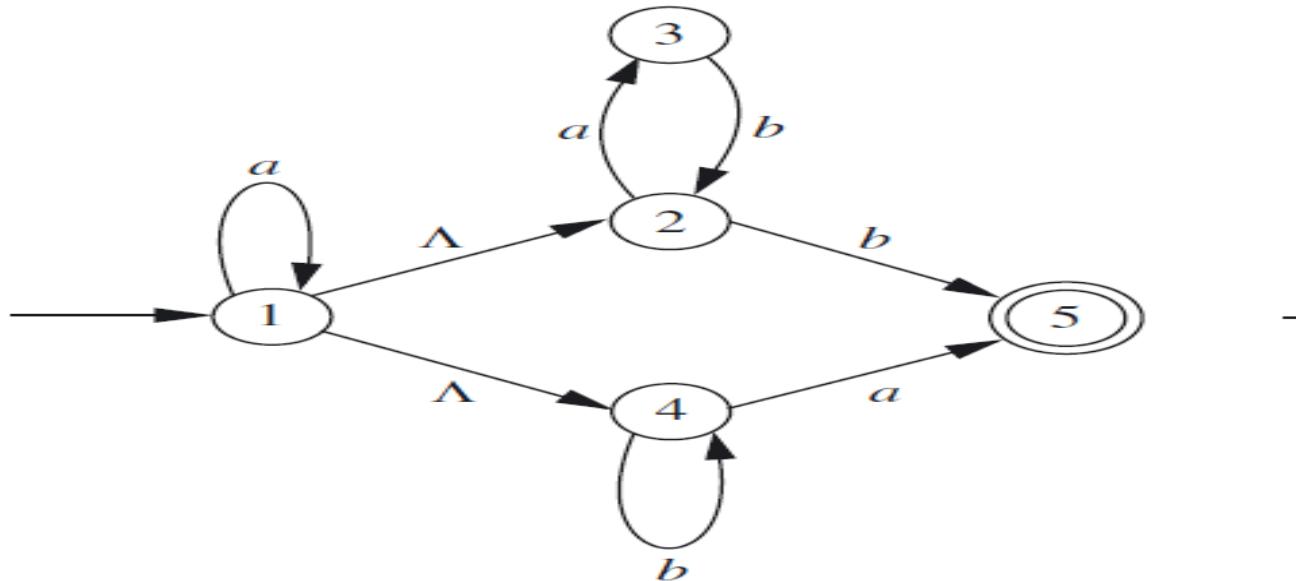
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$						



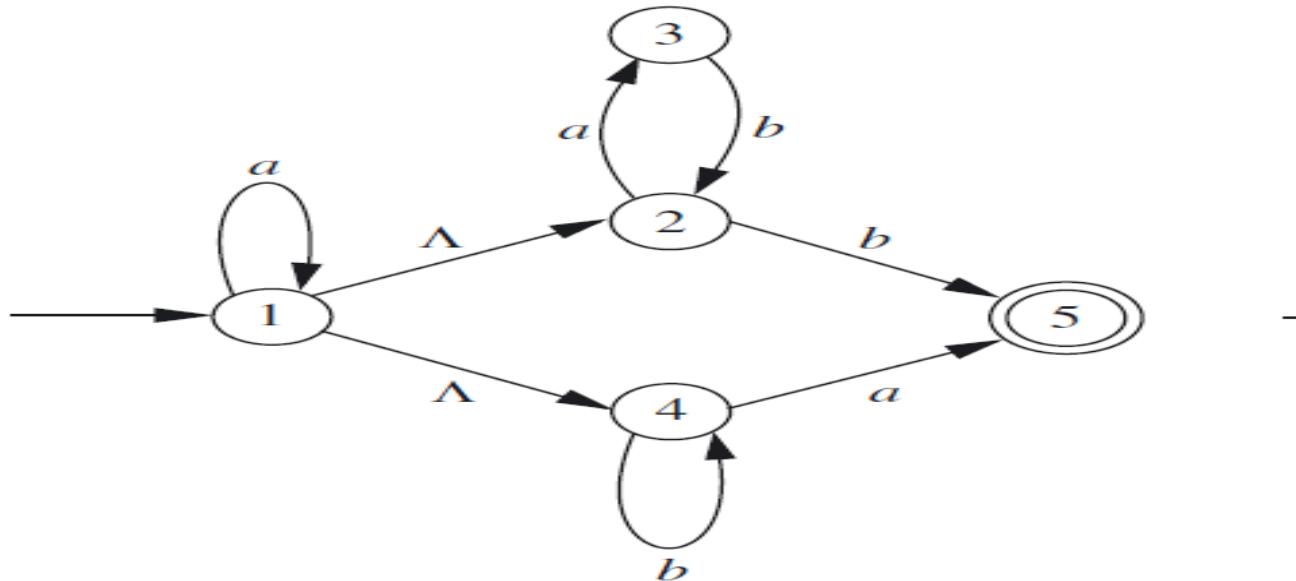
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$					



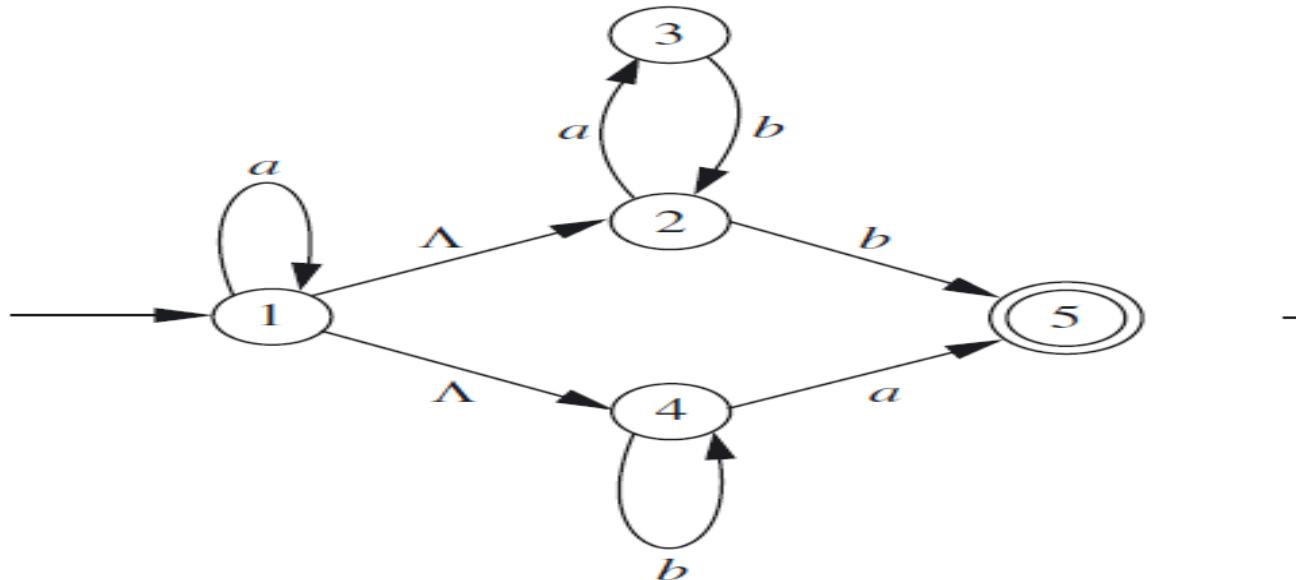
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$				



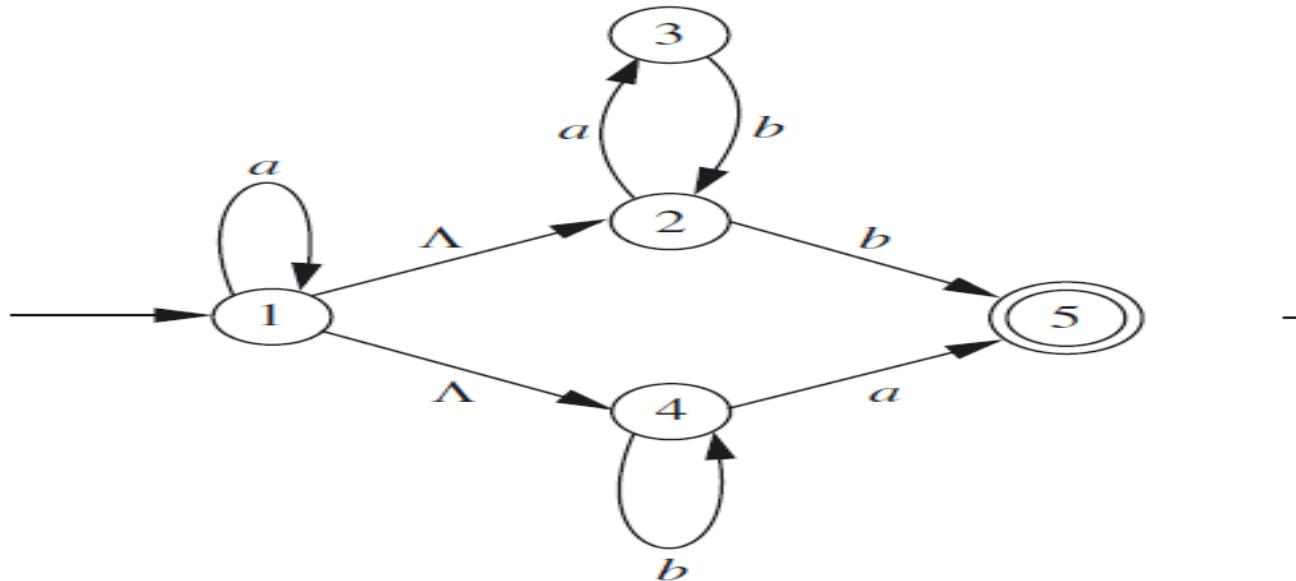
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$			



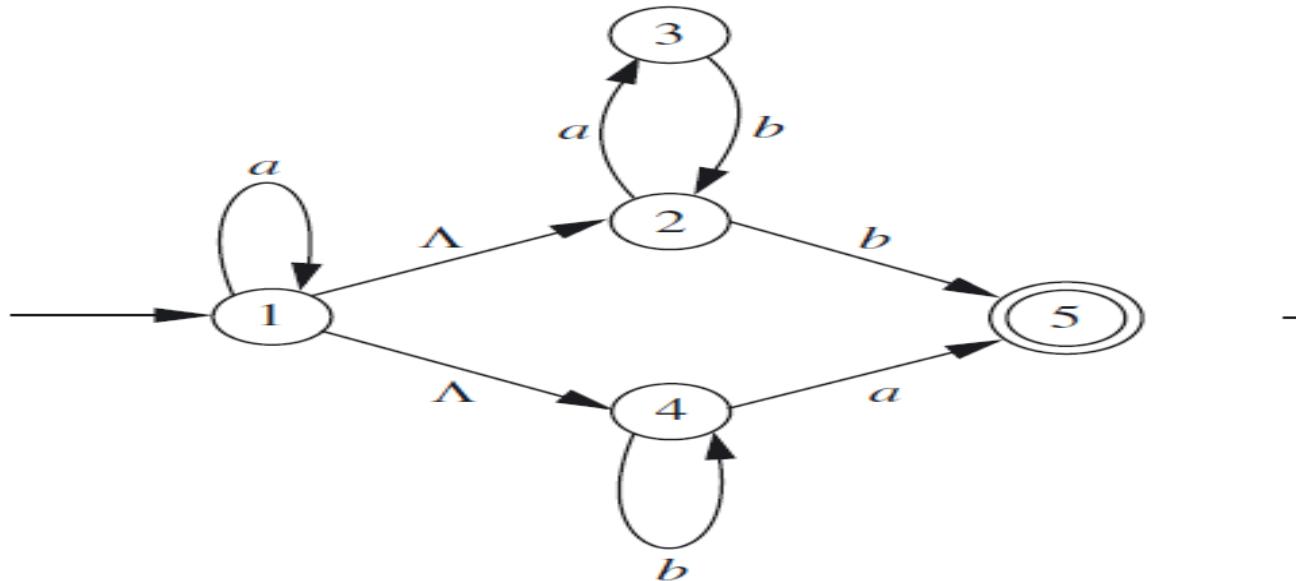
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	



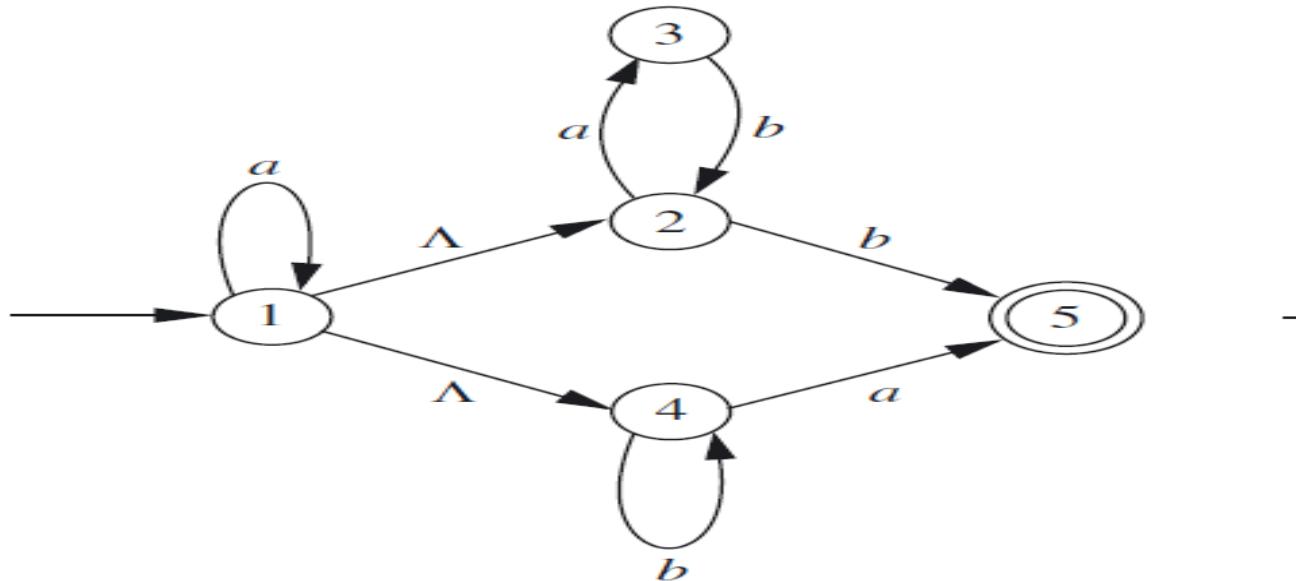
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$



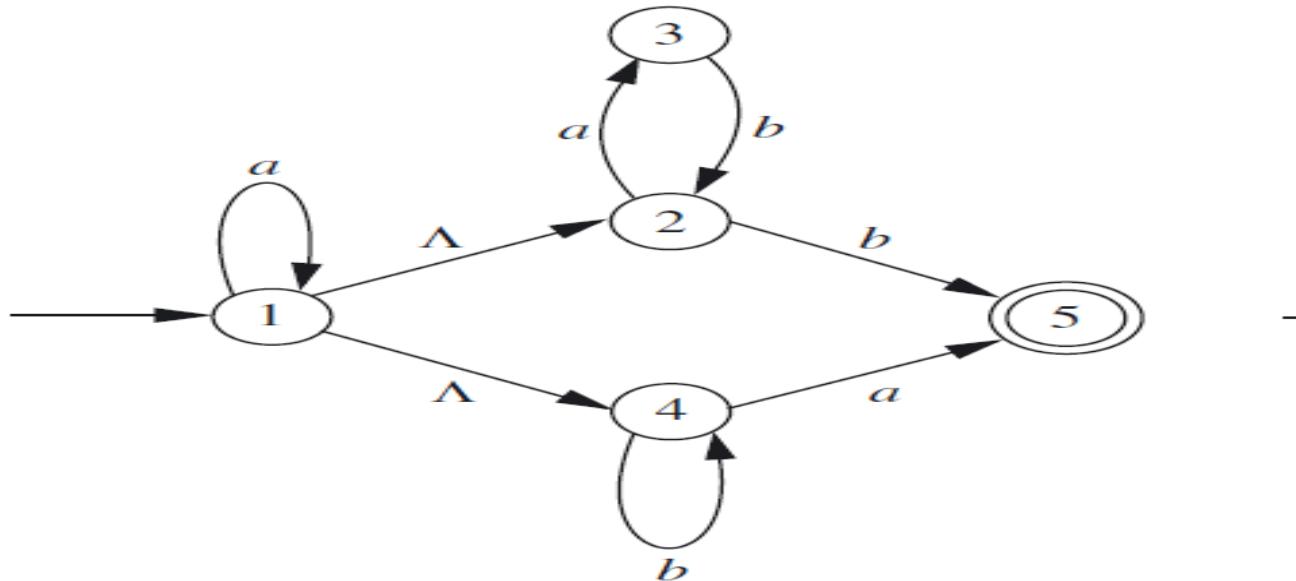
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$



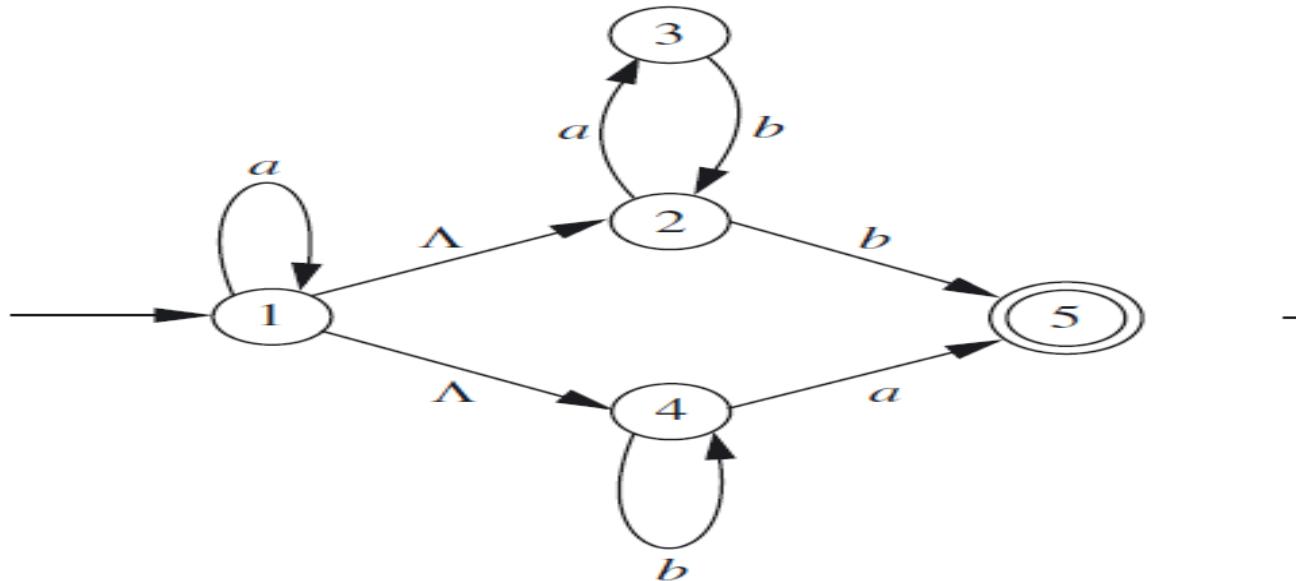
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4						



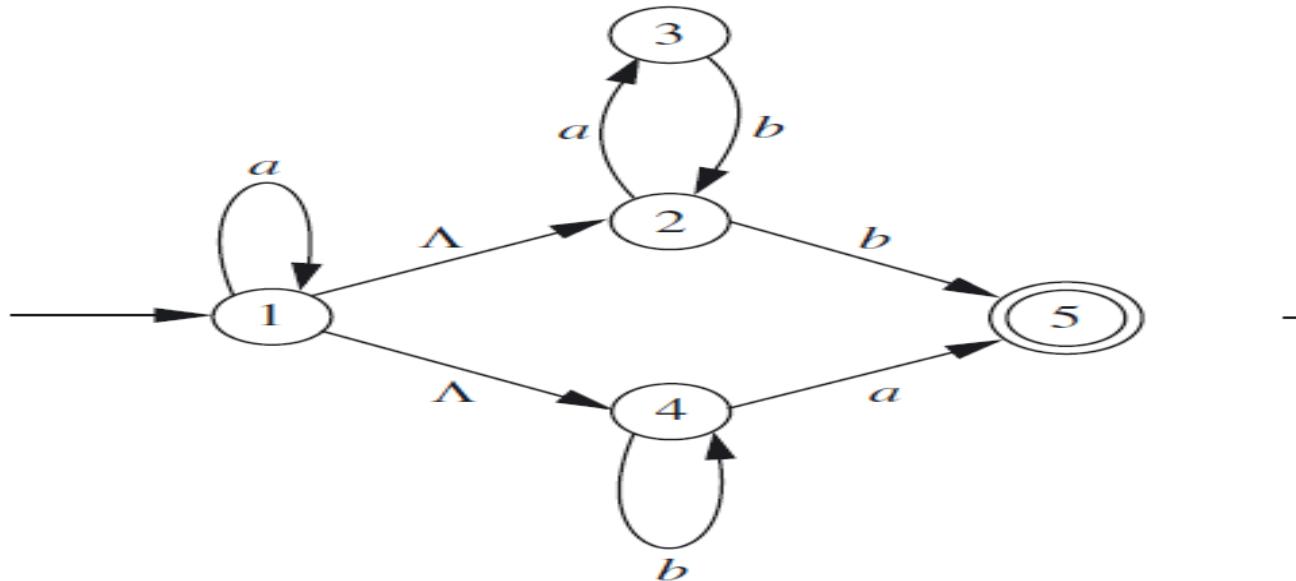
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$					



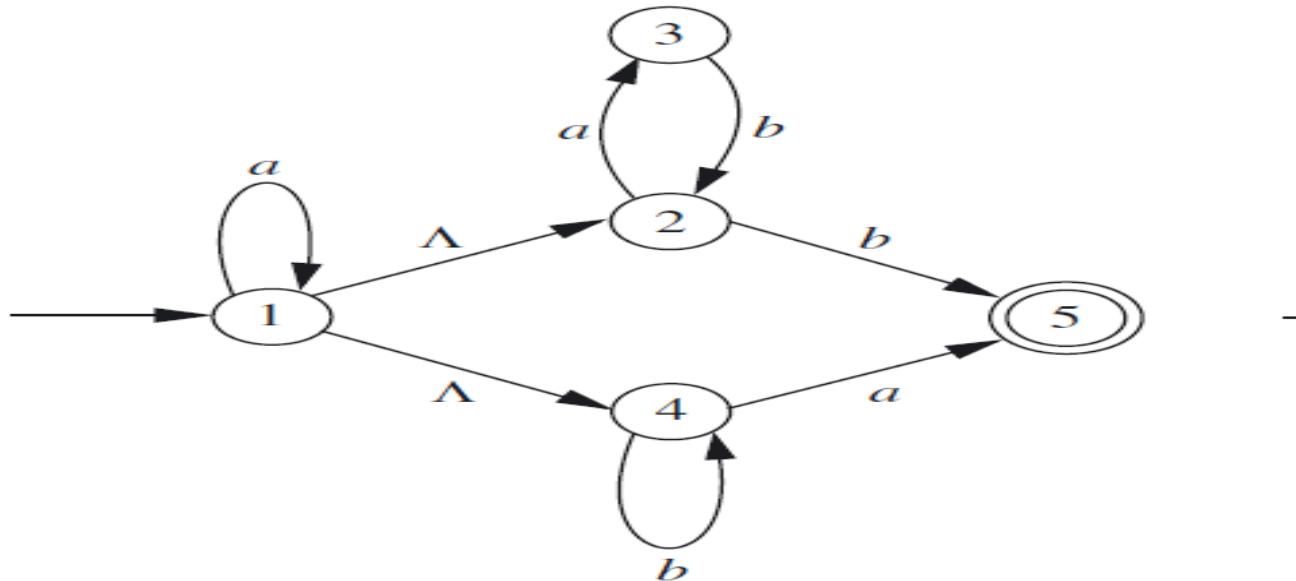
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$				



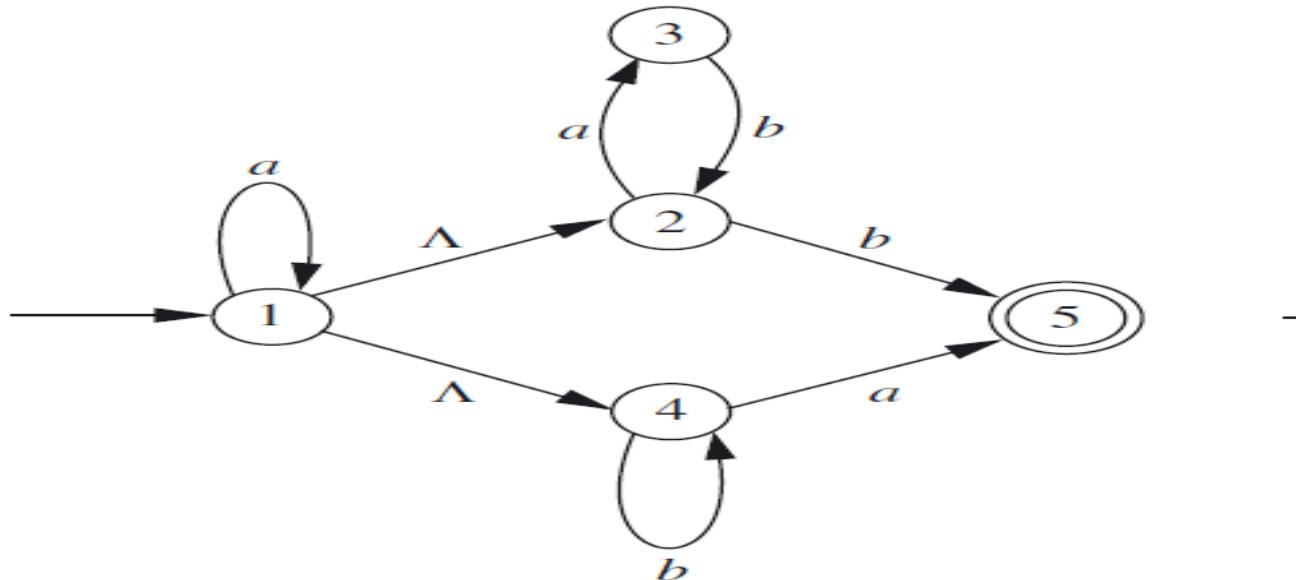
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$			



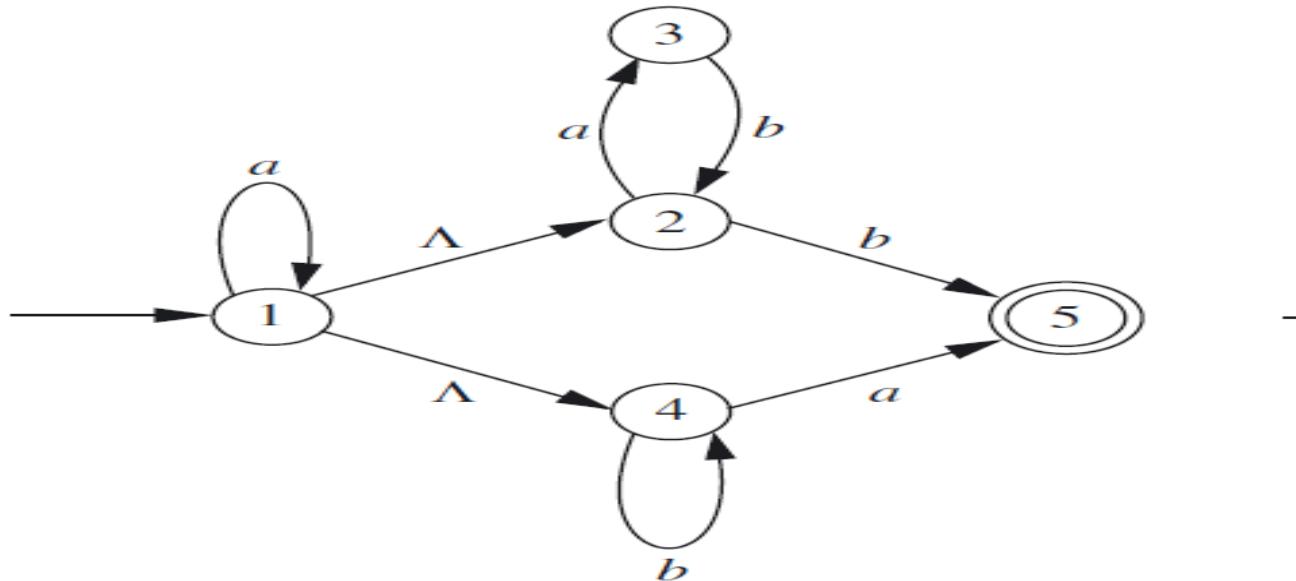
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	



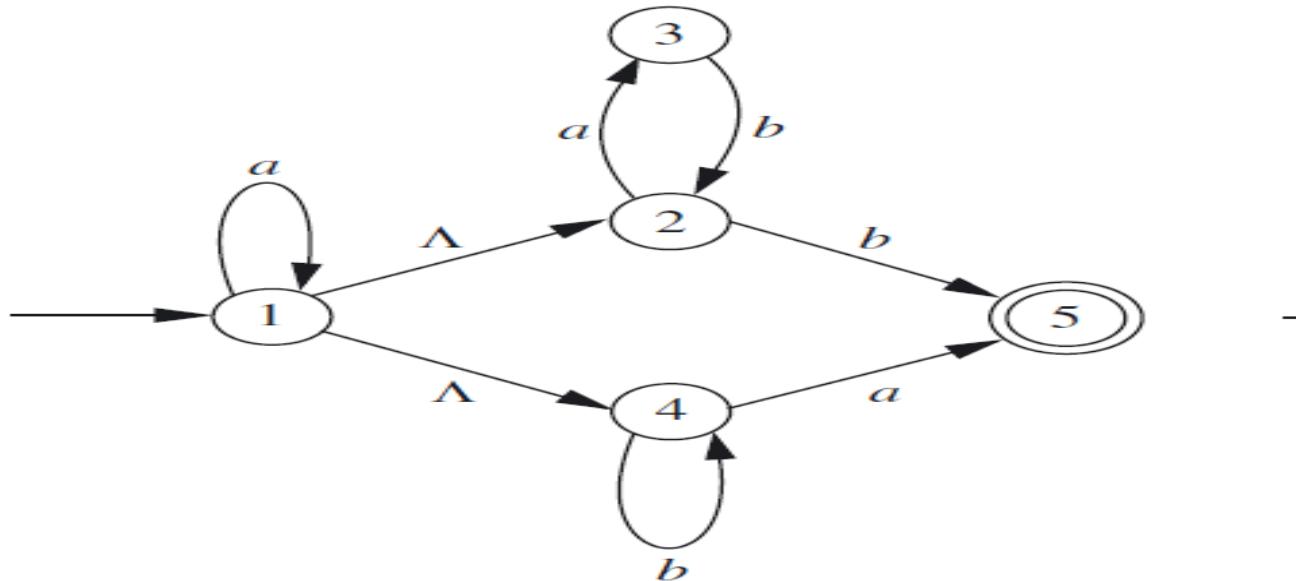
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$



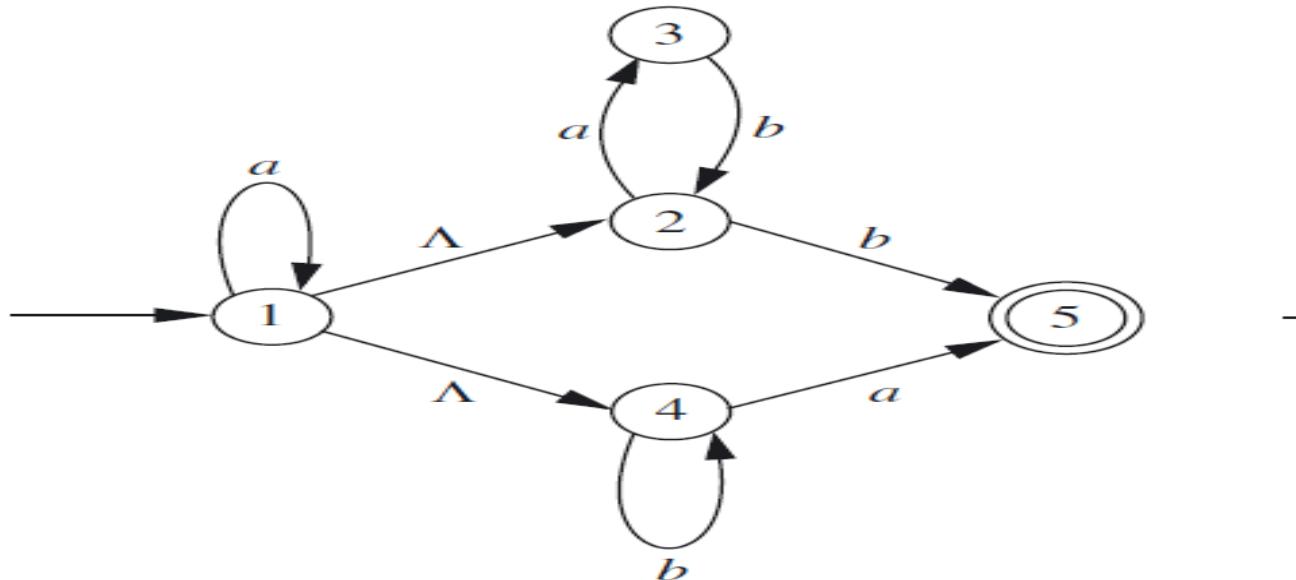
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$



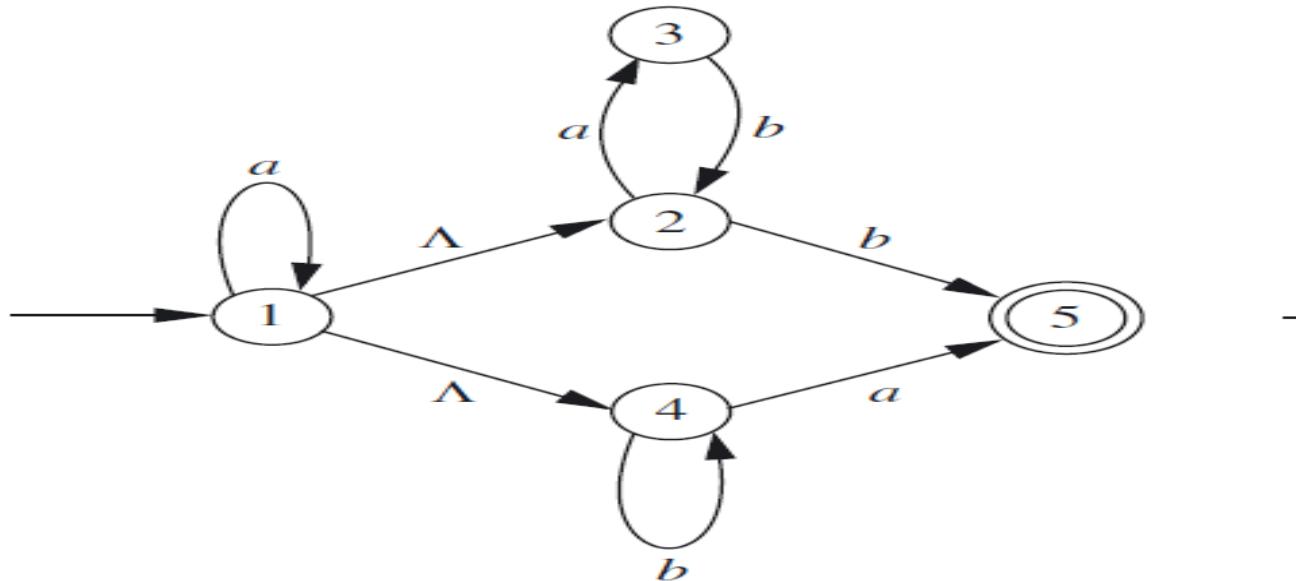
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5						



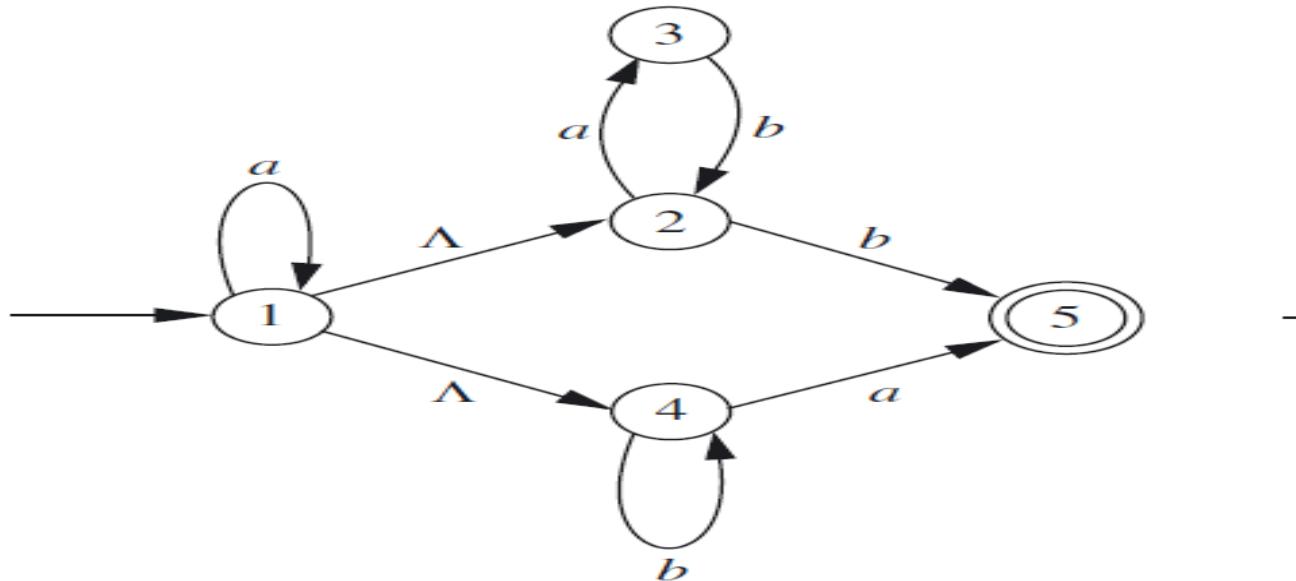
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$					



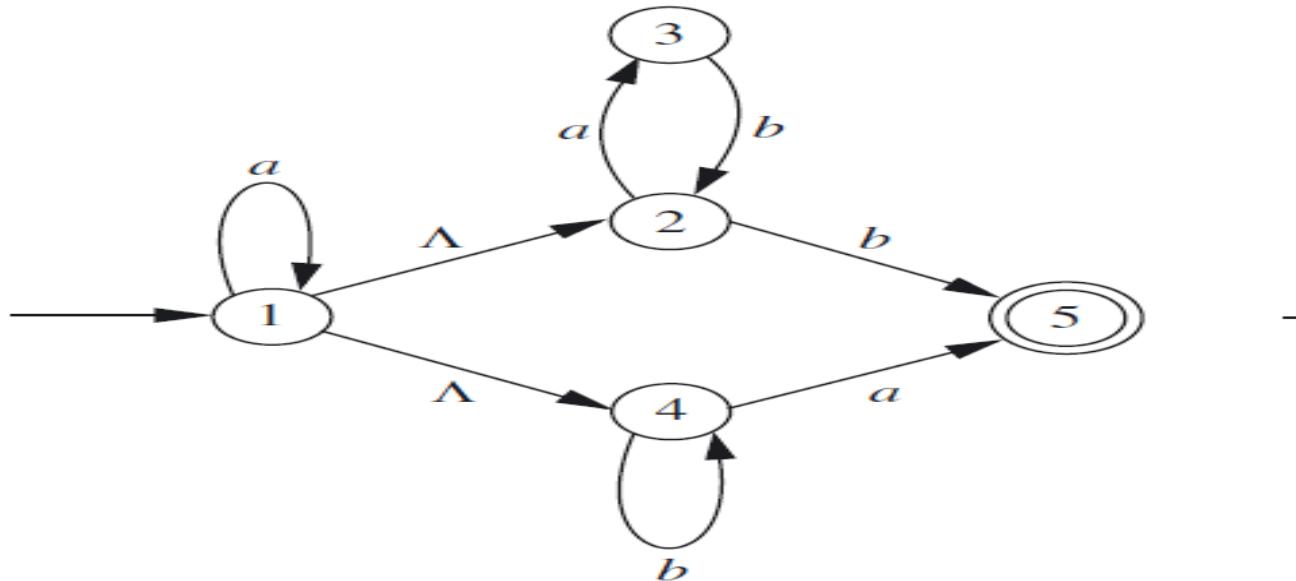
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$			



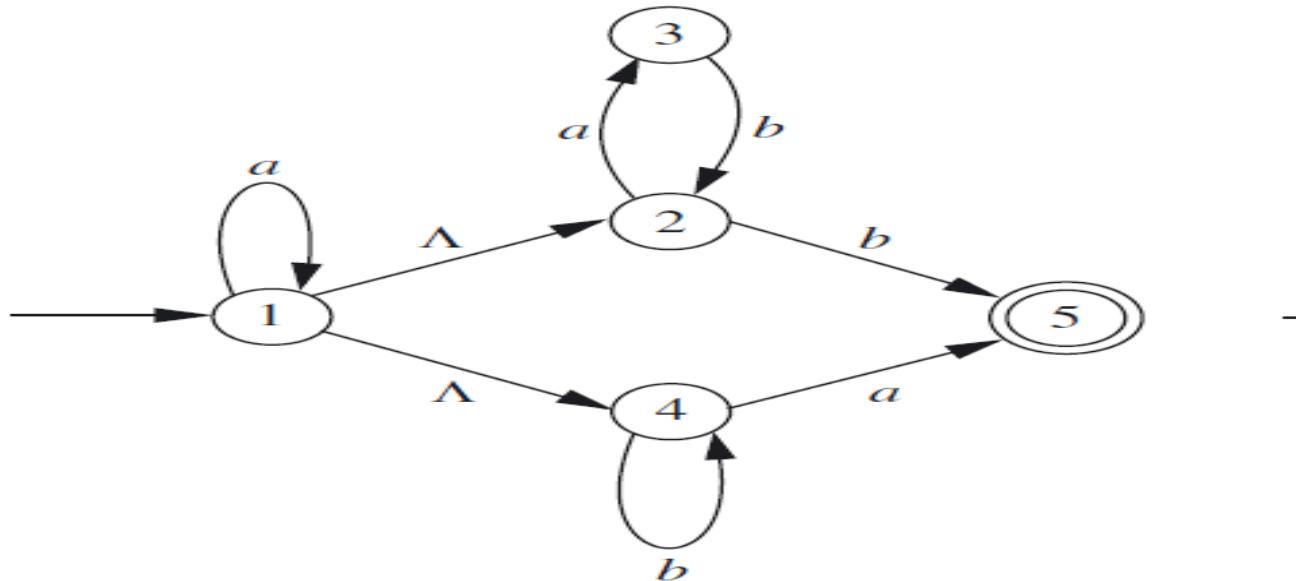
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$



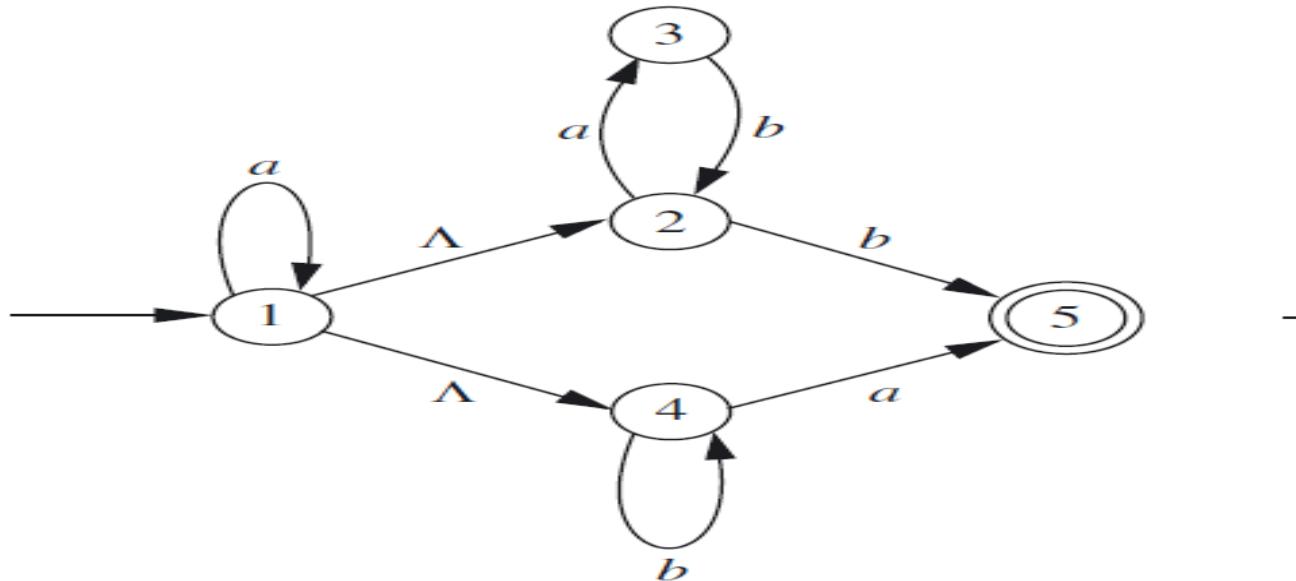
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$



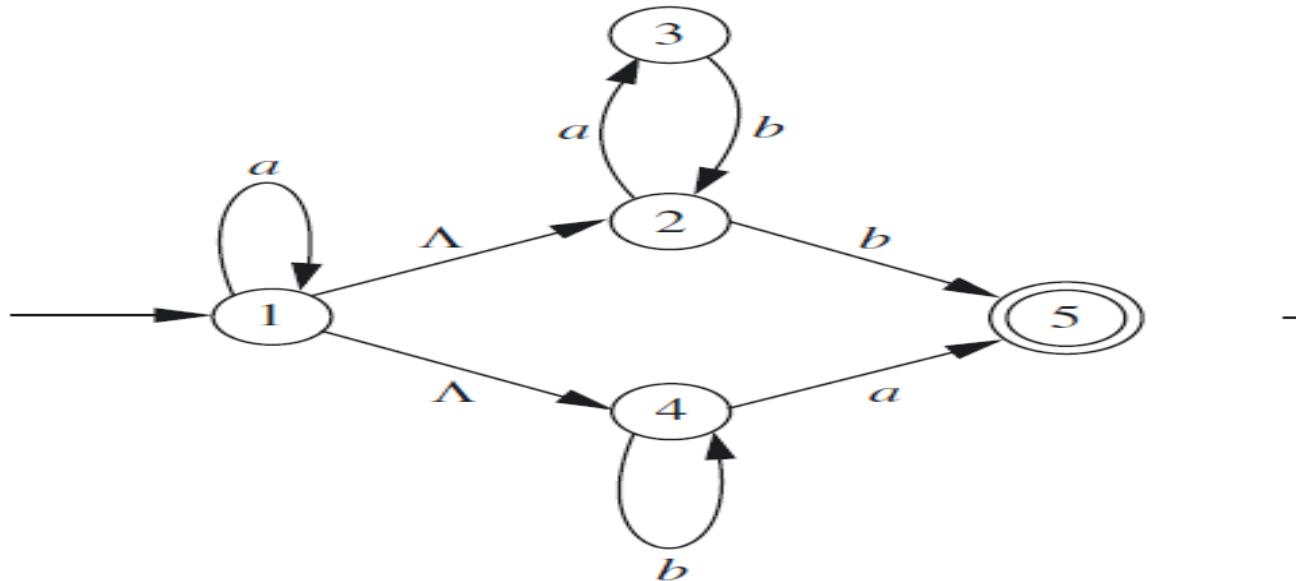
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$						



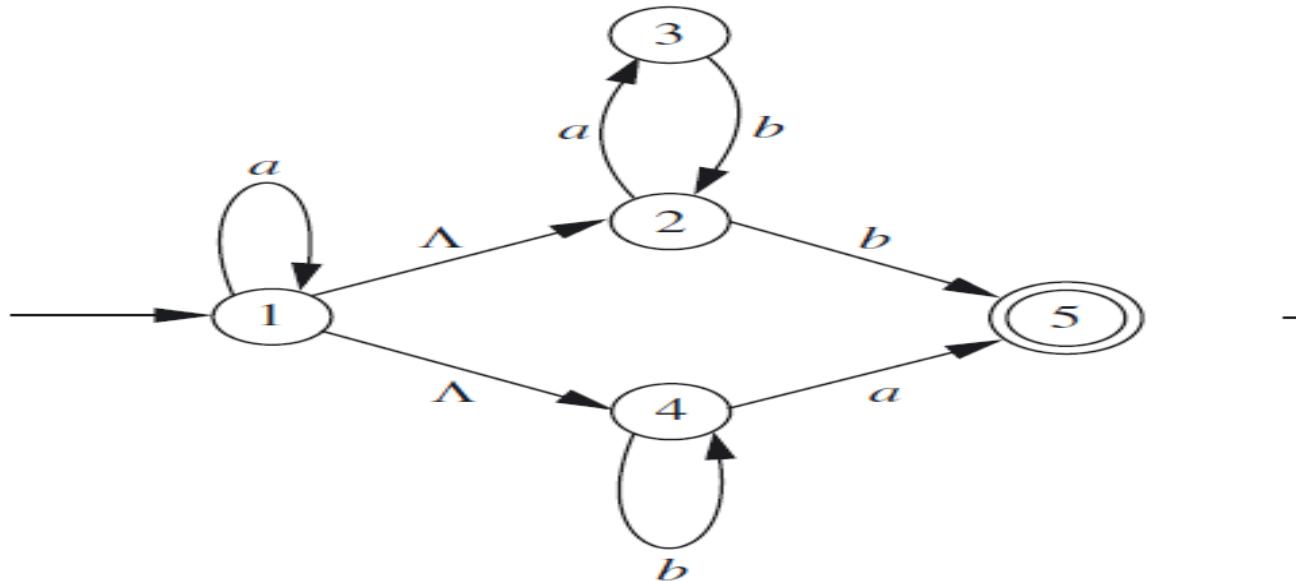
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$					



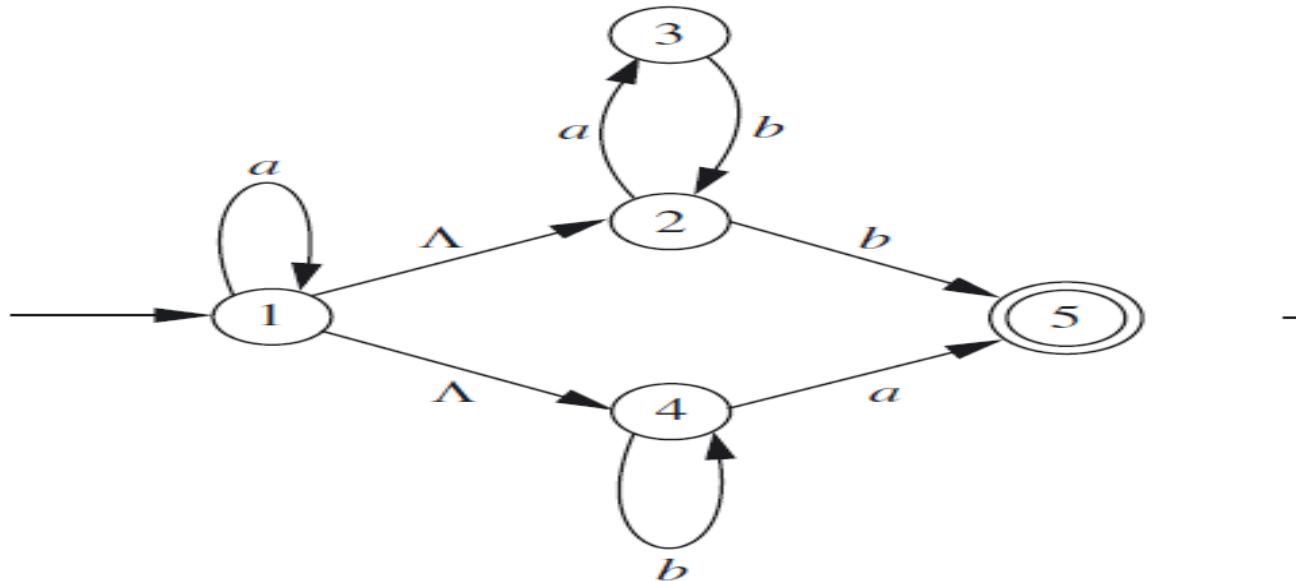
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$			



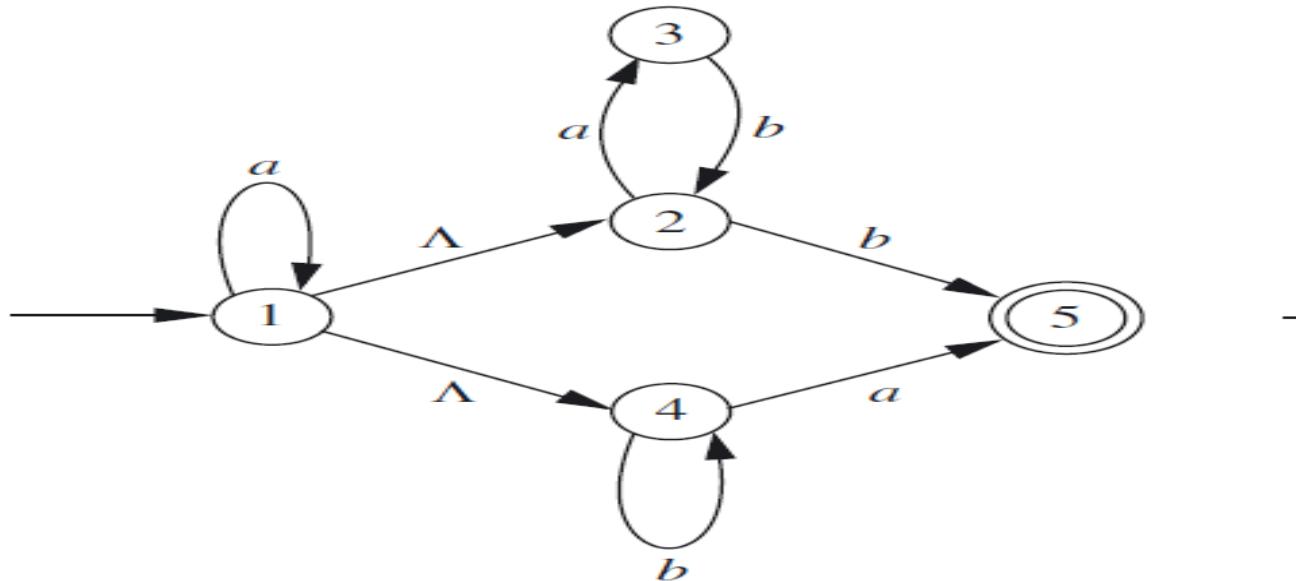
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	



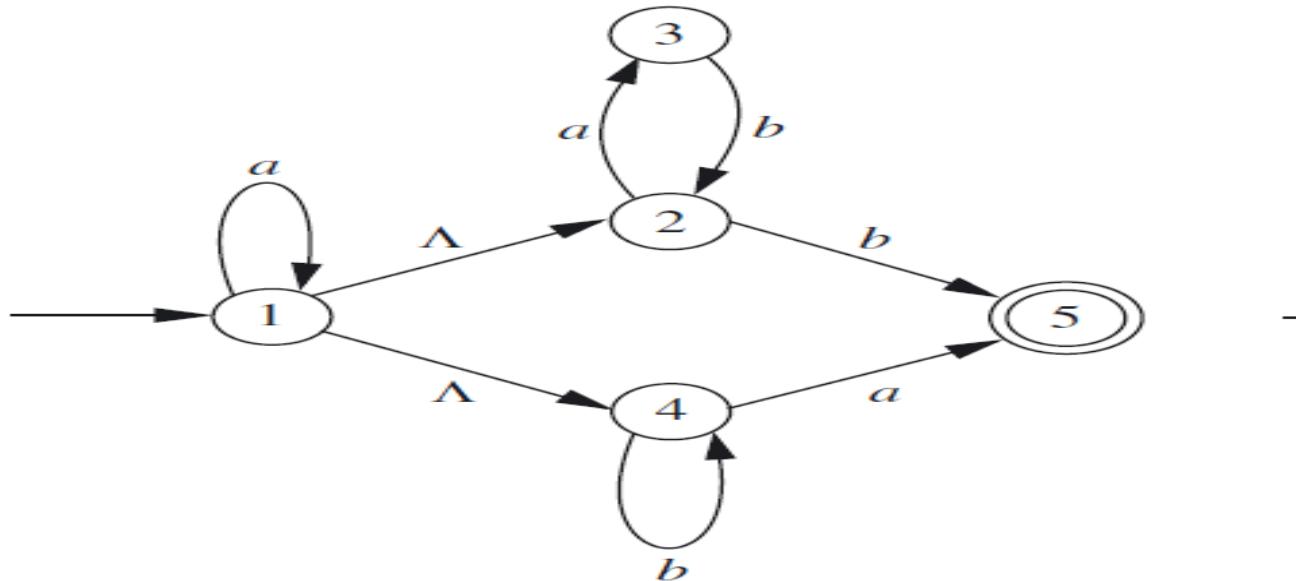
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$



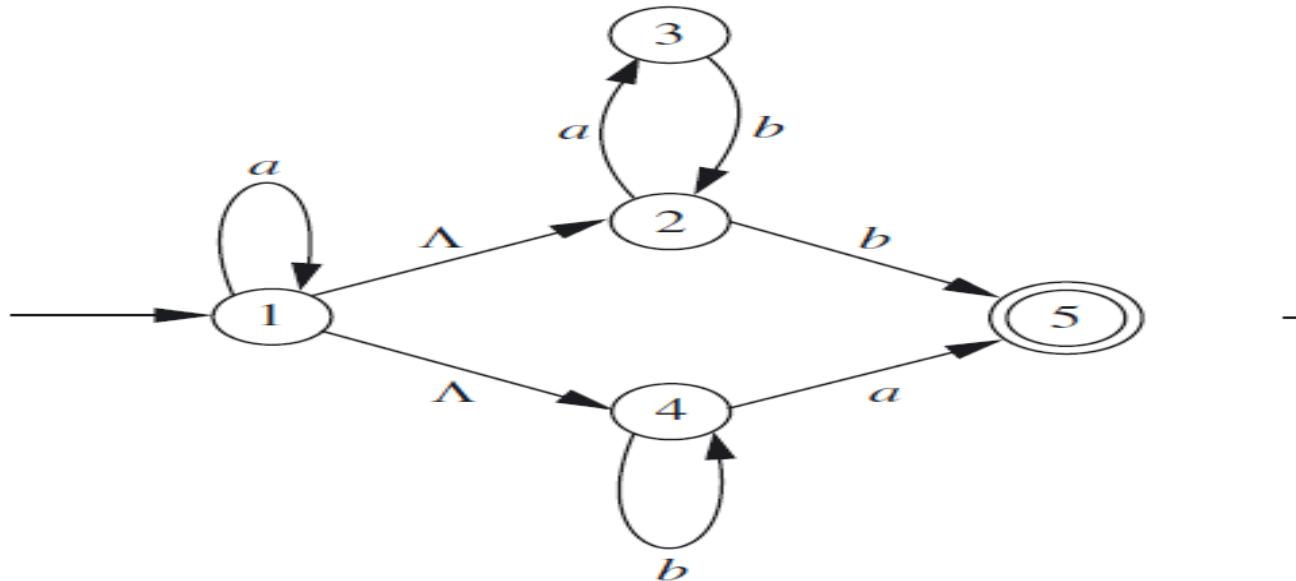
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$



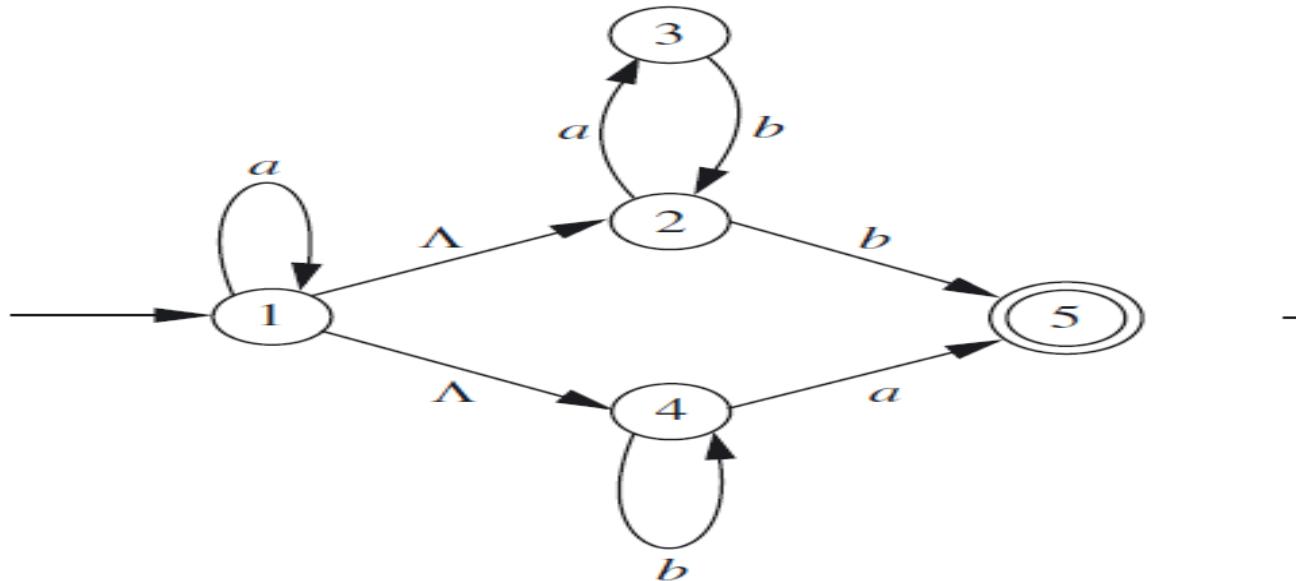
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2						



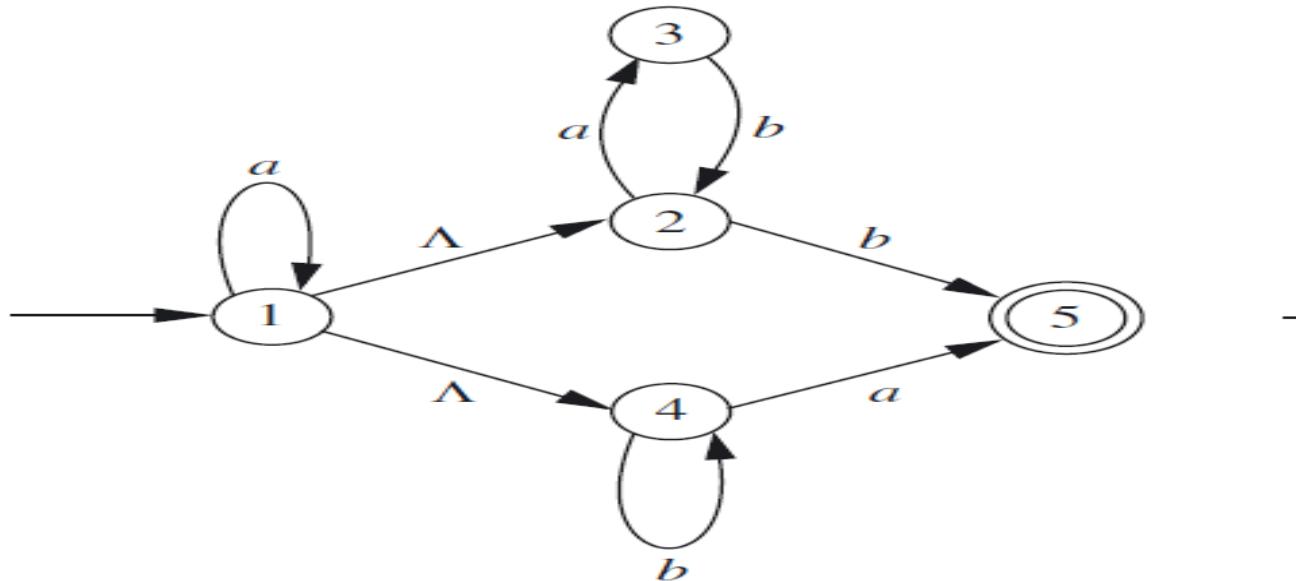
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$					



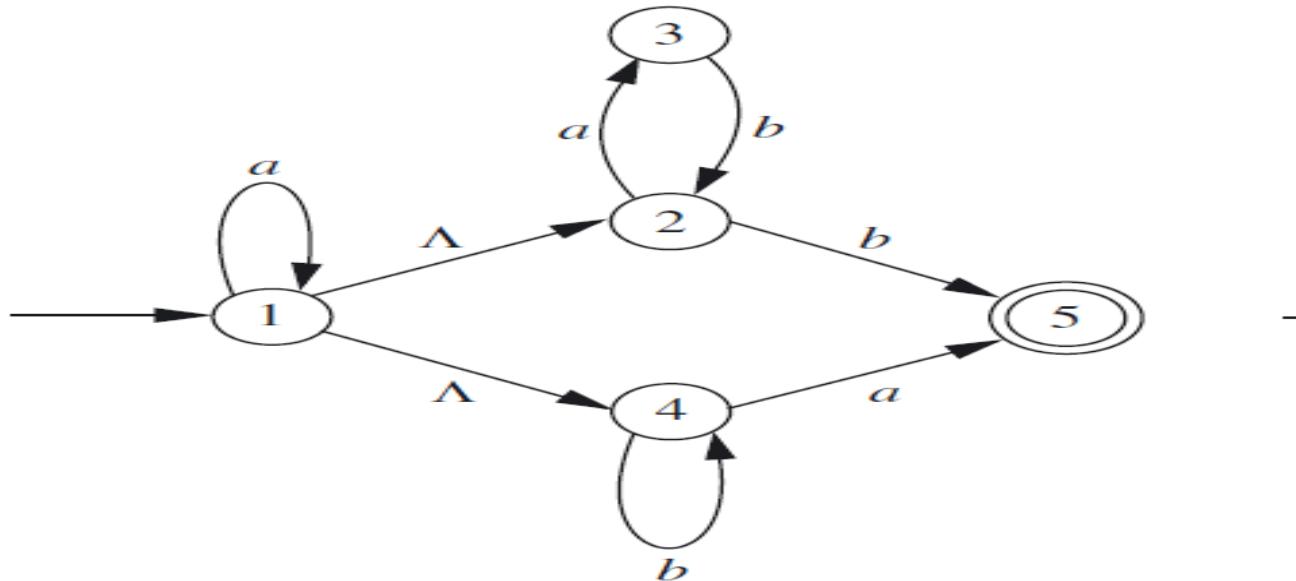
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$				



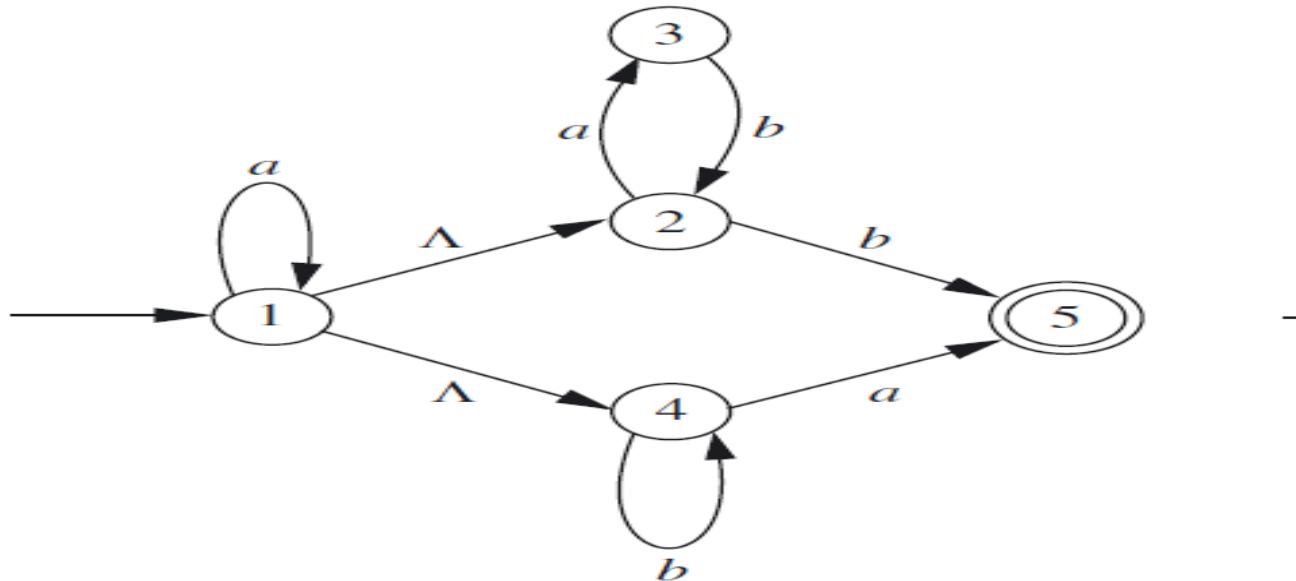
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$			



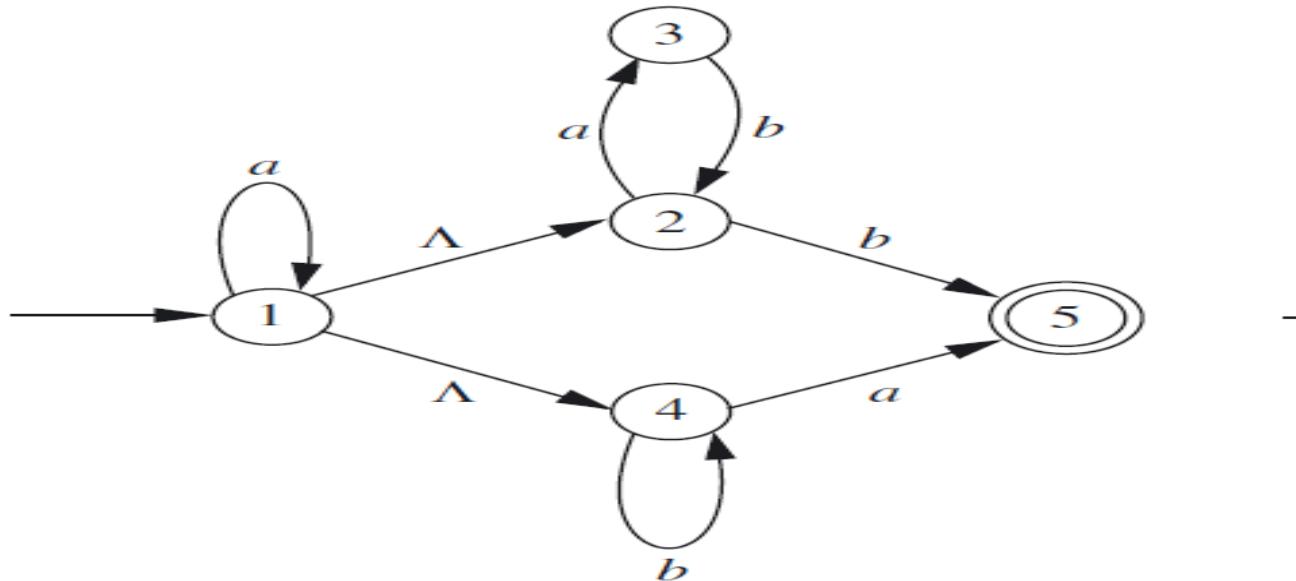
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$		



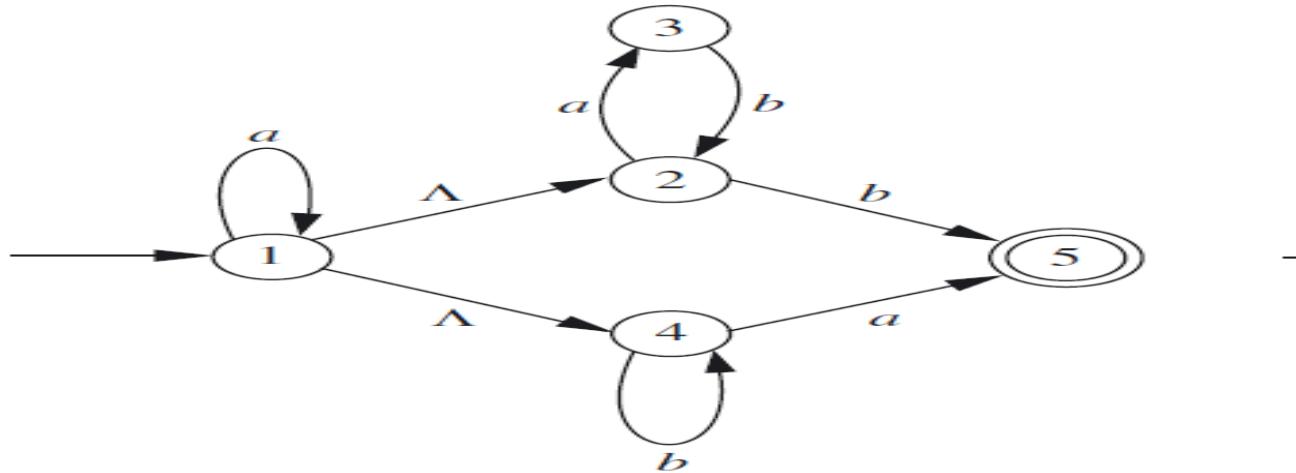
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	



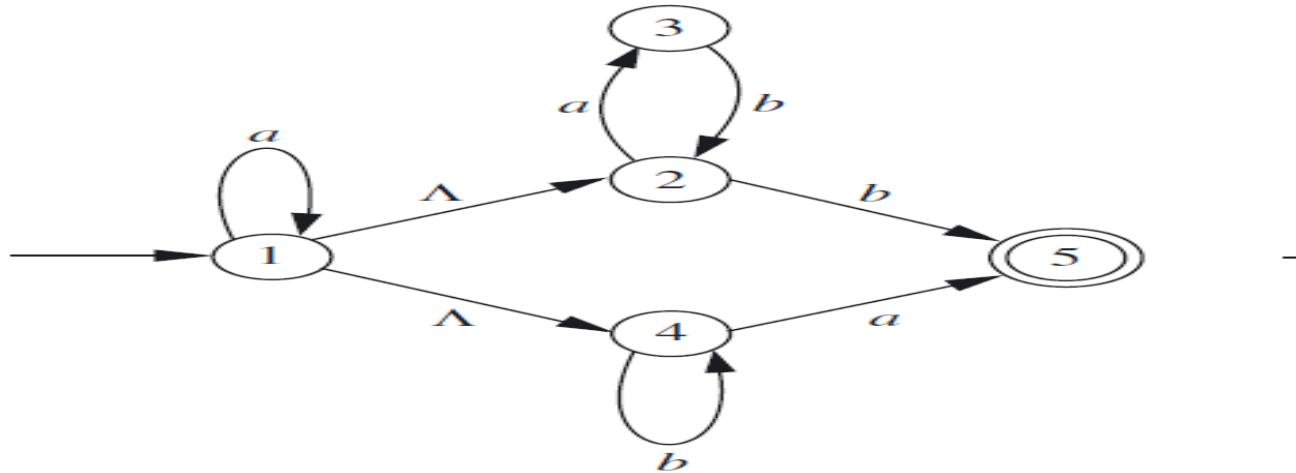
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$



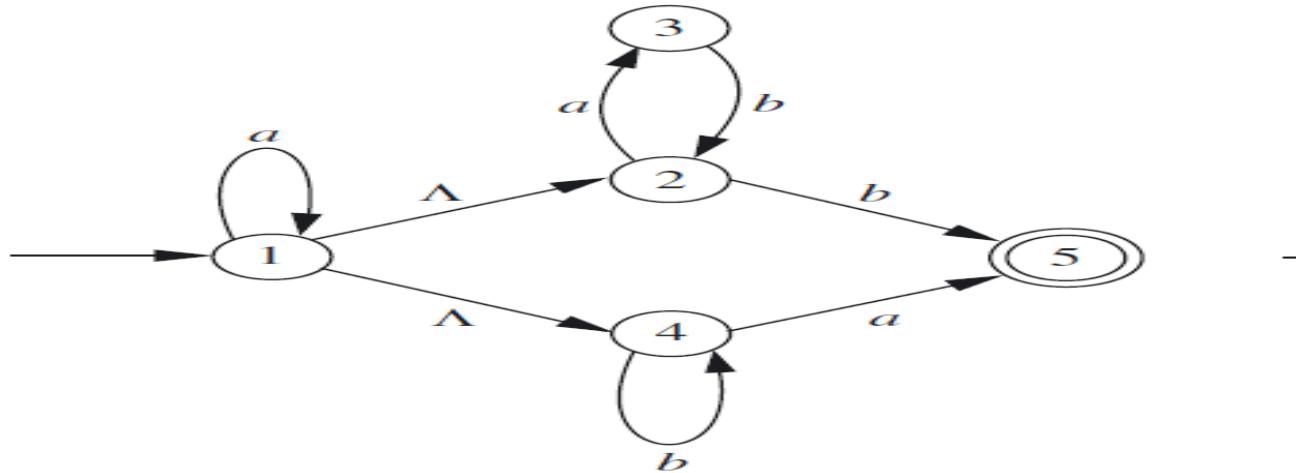
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$



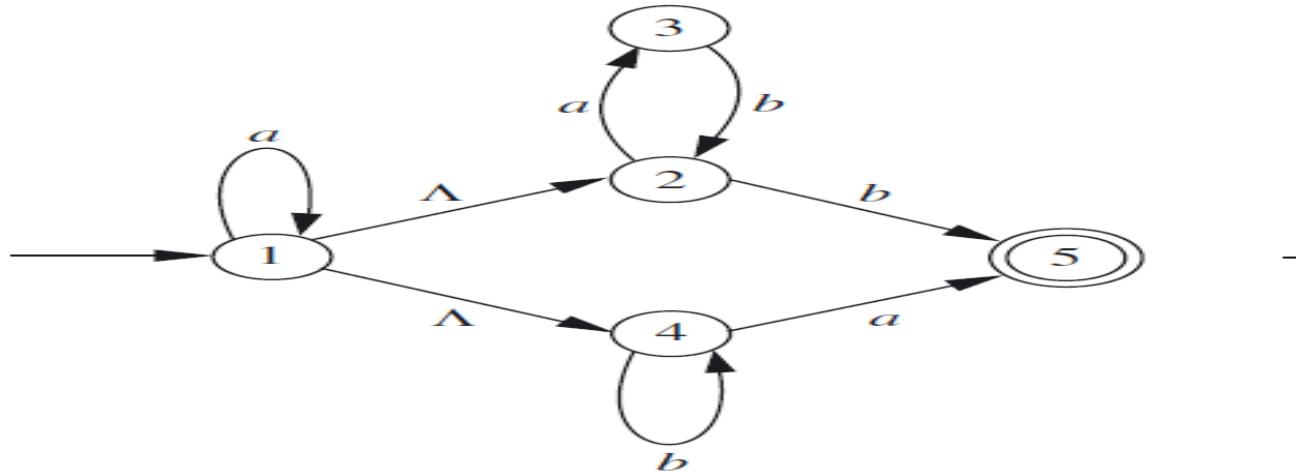
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
$\{3\}$						



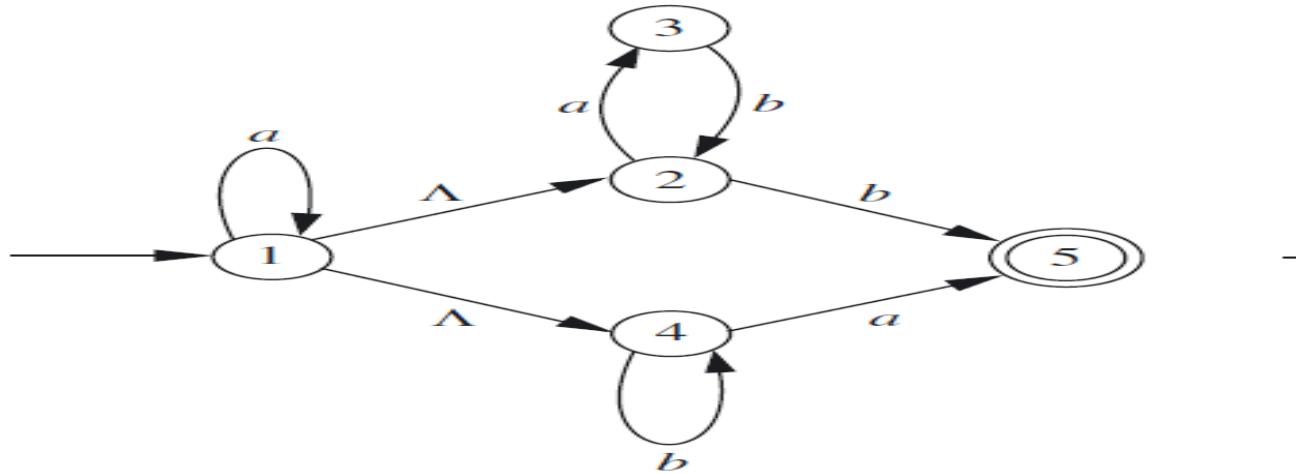
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
$\{3\}$	$\{3\}$					



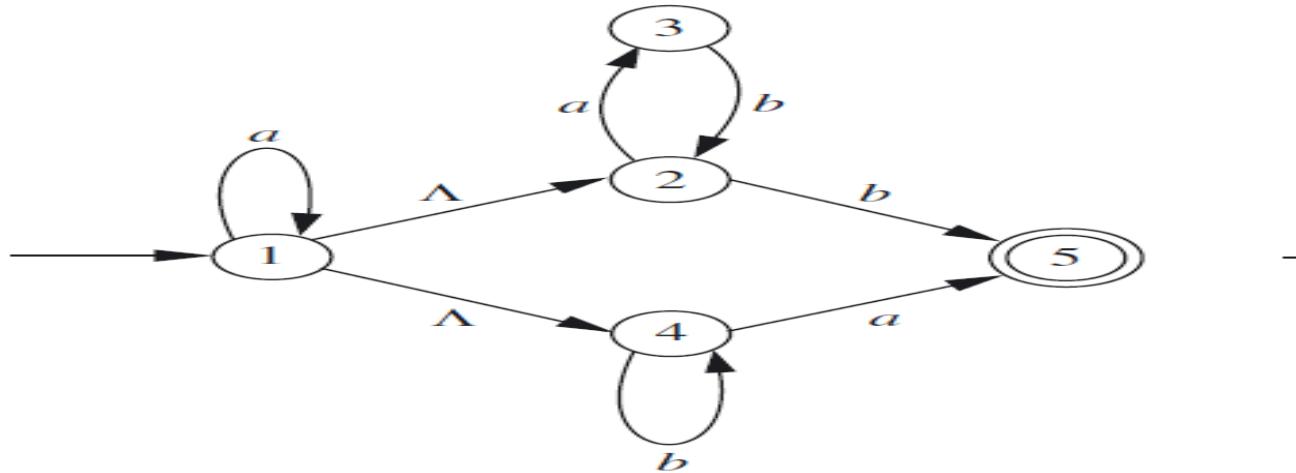
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
$\{3\}$	$\{3\}$	$\Phi$				



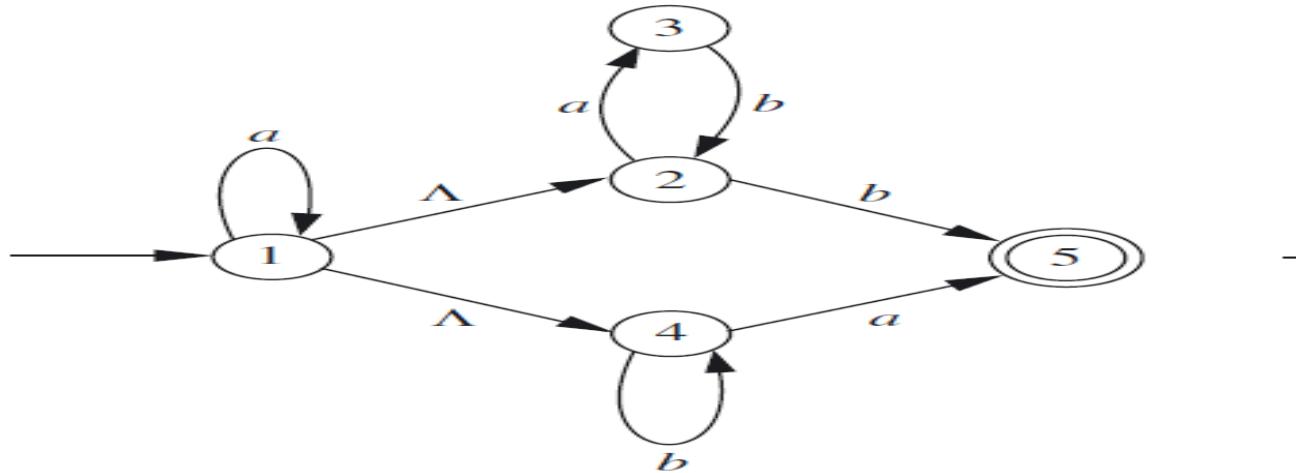
		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
$\{3\}$	$\{3\}$	$\Phi$	$\Phi$			



		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
$\{3\}$	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	



		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	2,4,5	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
$\{3\}$	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$

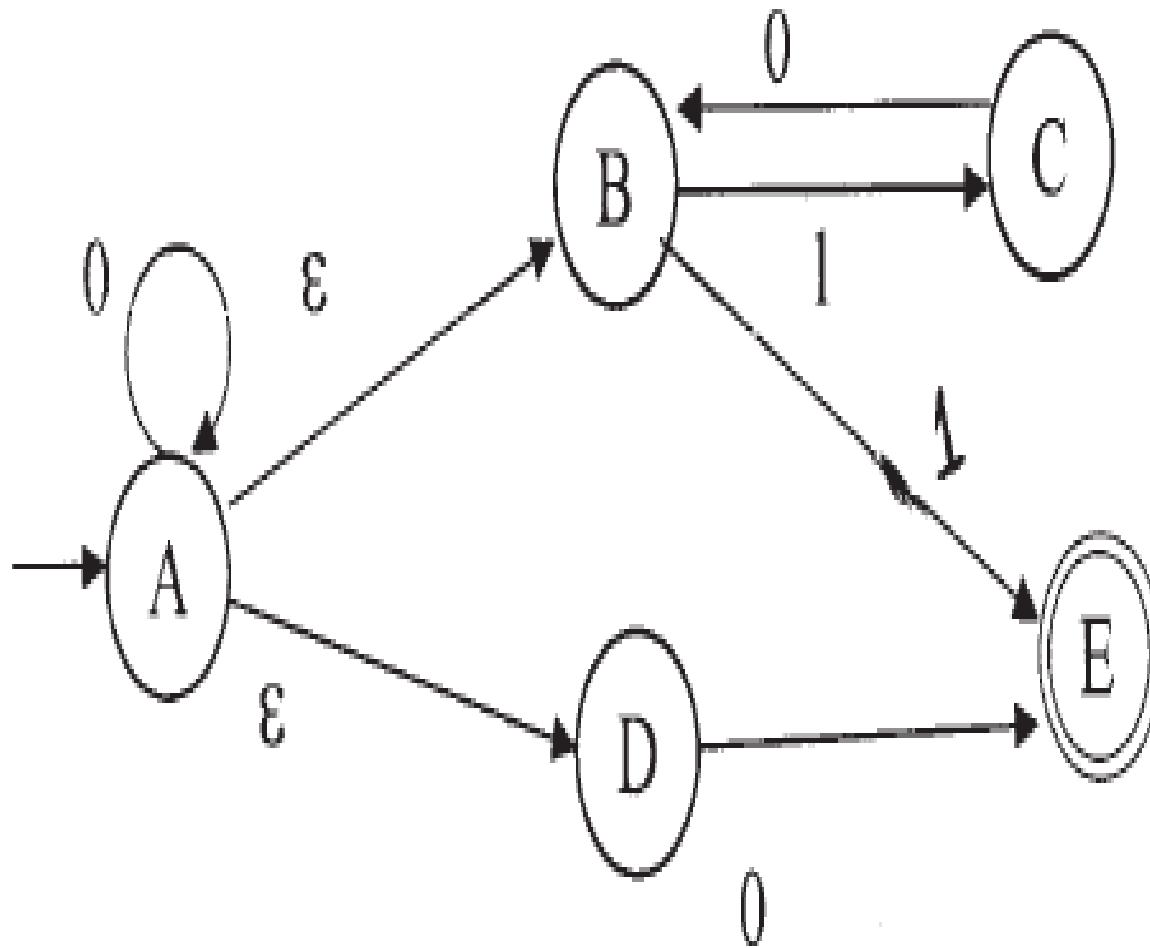


		$a$			$b$	
	$\lambda$		$\lambda$	$\lambda$		$\lambda$
1	$\{1,2,4\}$ ,	$\{1,3,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,4\}$	$\{4,5\}$
$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$\{1,2,3,4,5\}$	$2,4,5$	$\{2,4,5\}$
$\{4,5\}$	$\{4,5\}$	$\{5\}$	$\{5\}$	$\{4,5\}$	$\{4\}$	$\{4\}$
$\{2,4,5\}$	$\{2,4,5\}$	$\{3,5\}$	$\{3,5\}$	$\{2,4,5\}$	$\{4,5\}$	$\{4,5\}$
4	$\{4\}$	$\{5\}$	$\{5\}$	$\{4\}$	$\{4\}$	$\{4\}$
5	$\{5\}$	$\Phi$	$\Phi$	$\{5\}$	$\Phi$	$\Phi$
$\{3,5\}$	$\{3,5\}$	$\Phi$	$\Phi$	$\{3,5\}$	$\{2\}$	$\{2\}$
2	$\{2\}$	$\{3\}$	$\{3\}$	$\{2\}$	$\{5\}$	$\{5\}$
$\{3\}$	$\{3\}$	$\Phi$	$\Phi$	$\{3\}$	$\{2\}$	$\{2\}$

02/07/2019Absent

- 3,16,17,27,33,57,62,71,74

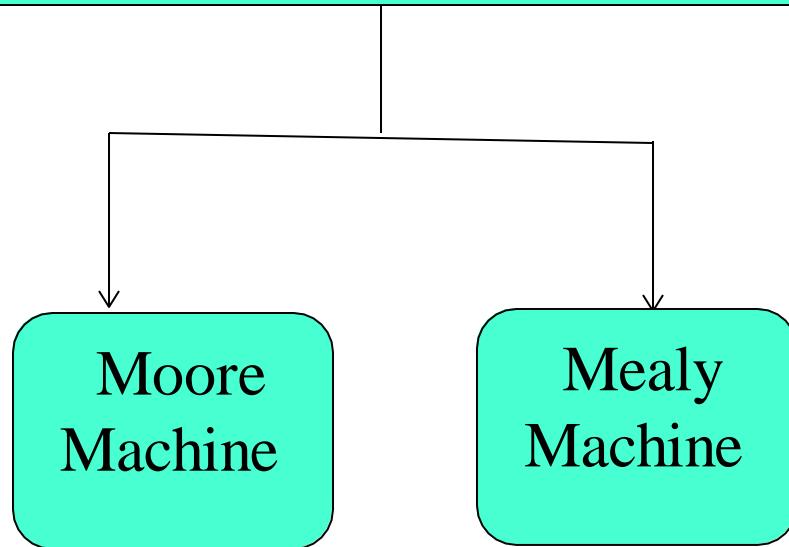
Convert the given NFA- $\epsilon$  to an NFA.



# Review

- DFA, NFA and NFA- $\lambda$  Equivalency
- Conversion of NFA to DFA, NFA- $\lambda$  to DFA

## Finite Automata with o/p



# • Moore Machine and Mealy machine

Definition:-

$$(Q, \Sigma, q_0, \delta, \Delta, \gamma)$$

Where,

$Q$ =set of states,  $\Sigma$ =i/p alphabet,  $q_0$ = start/initial state,

$\delta=Q \times \Sigma \rightarrow Q$  (transition function),

$\Delta$ =o/p alphabet,  $\gamma$  =o/p function

• Difference between Mealy and Moore machine is  $\gamma$  (o/p function)

$\gamma$  (o/p function) for Moore machine is  $\gamma : Q \rightarrow \Delta$

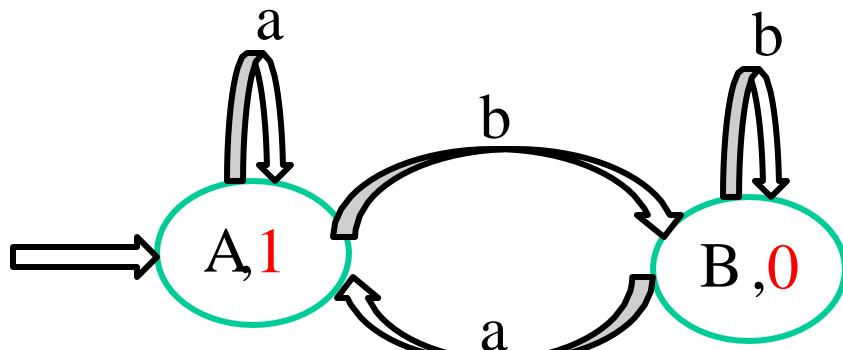
(outputs depend on only the present state)

$\gamma$ (o/p function) for Mealy machine is  $\gamma : Q \times \Sigma \rightarrow \Delta$

(Output depends on the present state as well as the present input)

• Moore and Mealy Machines are equally powerful

- Moore Machine



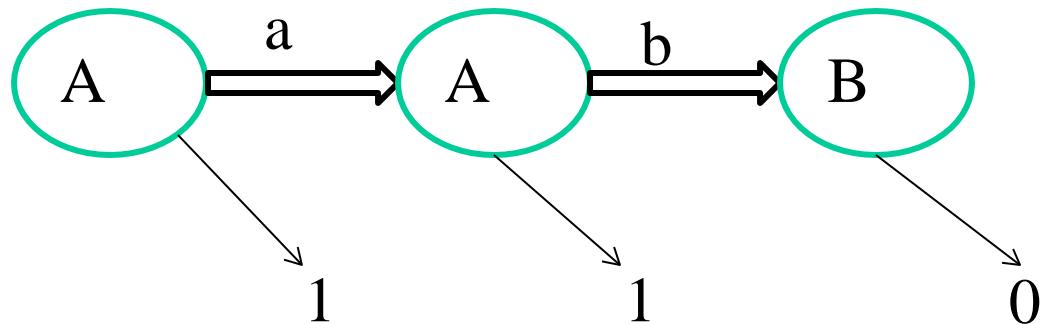
	Input		Output
	a	b	
A	A	B	1
B	A	B	0

$$\lambda: Q \rightarrow \Delta$$

$A \rightarrow 1$

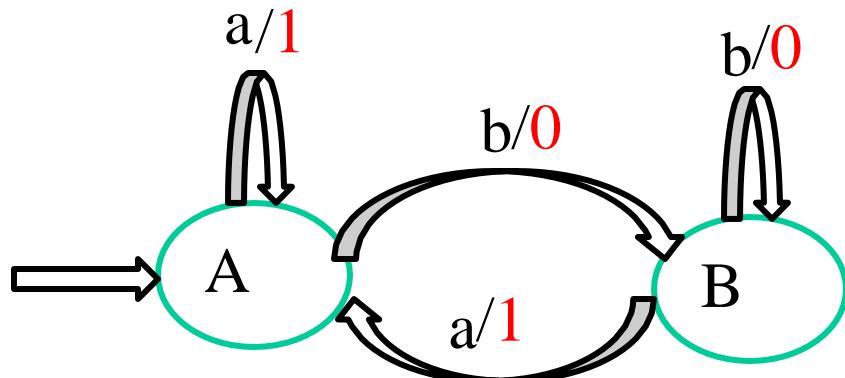
$B \rightarrow 0$

- string 'ab' on machine



- Applying string 'ab' on machine (as input), got the output as 110.
- If number of input symbol is N then Number of output symbol is N+1.

## • Mealy Machine



	Input	Output	Input	Output
	a		b	
A	A	1	B	0
B	A	1	B	0

$$\lambda: Q \times \Sigma \rightarrow \Delta$$

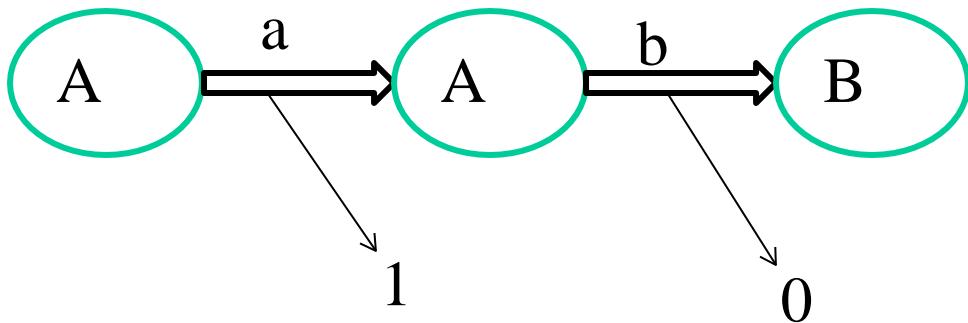
$$A, a \rightarrow 1$$

$$A, b \rightarrow 0$$

$$B, a \rightarrow 1$$

$$B, b \rightarrow 0$$

• string 'ab' on machine

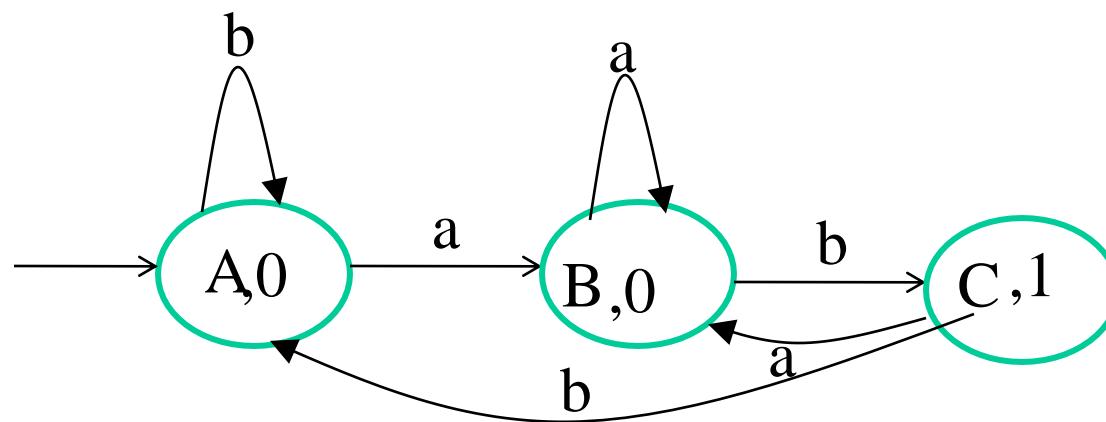


- Applying string 'ab' on machine (as input), got the output as 10.
- If number of input symbol is N then Number of output symbol is N.

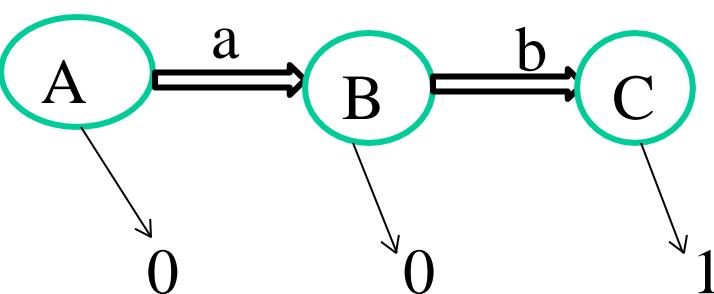
## Example on Moore Machine

Construct a Moore Machine that takes the set of all strings over  $\{a, b\}$  as i/p and prints '1' as o/p for every occurrence of 'ab' as a substring

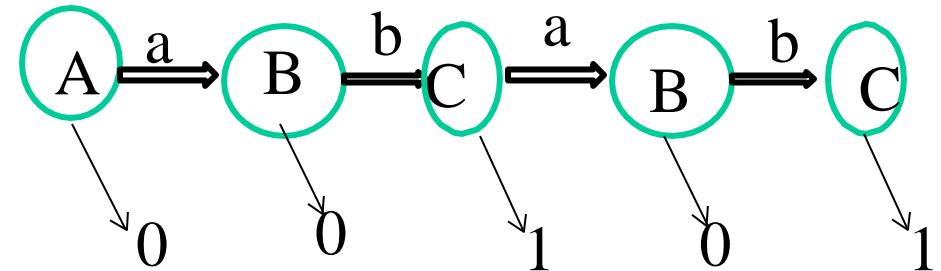
$$Q = \{A, B, C\} \quad \Sigma = \{a, b\} \quad \Delta = \{0, 1\}$$



• string 'ab' on machine



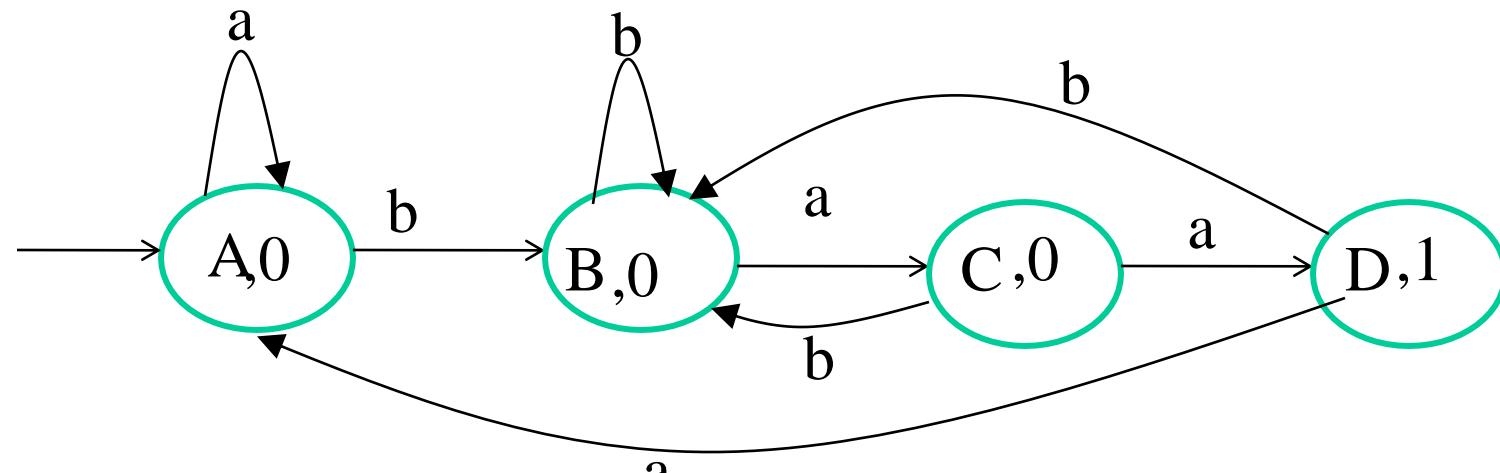
• string 'abab' on machine



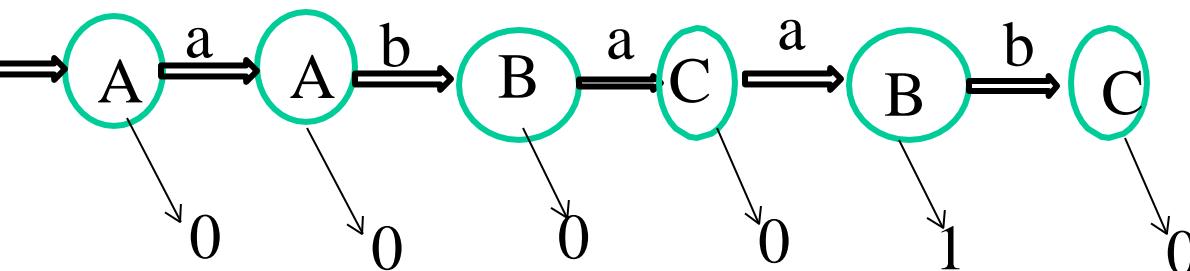
## Example on Moore Machine

Construct a Moore Machine that takes the set of all strings over  $\{a, b\}$  as i/p and counts no of occurrences of substring 'baa'

$$Q = \{A, B, C\} \quad \Sigma = \{a, b\} \quad \Delta = \{0, 1\}$$



• string 'abaab' on machine



## Example on Moore Machine

Construct a Moore Machine that takes the set of all strings over  $\{0, 1\}$  and produces 'A' as o/p if i/p ends with '10' or produces 'B' as o/p if i/p ends with '11' otherwise produces 'C'

## Example on Moore Machine

Construct a Moore Machine that takes binary no's as input and produces residue modulo '3' as output.

## Example on Mealy Machine

Construct a Mealy Machine for 1's complement of binary number.

## Example on Mealy Machine

Construct a Mealy Machine that takes binary number as i/p and produces 2's complement of that number as o/p. Assume the string is read LSB to MSB and end carry is discarded.

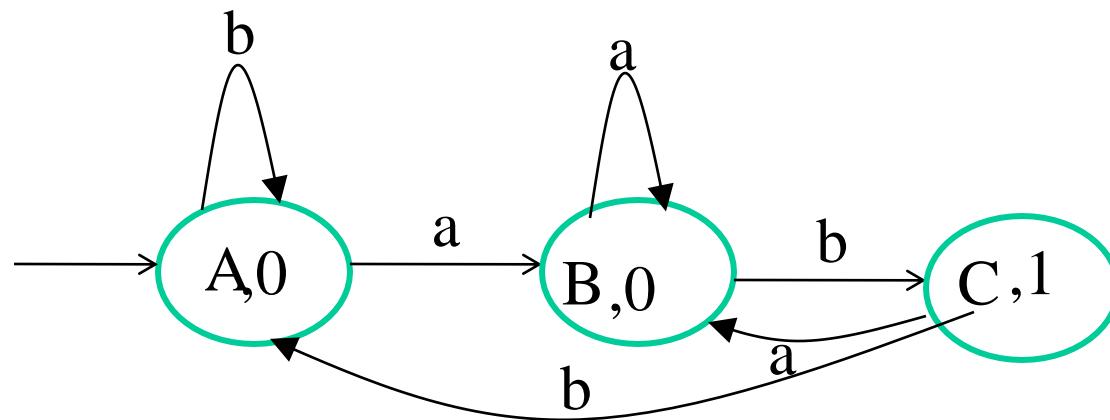
# Recall

- FA With O/P
- Moore Machine

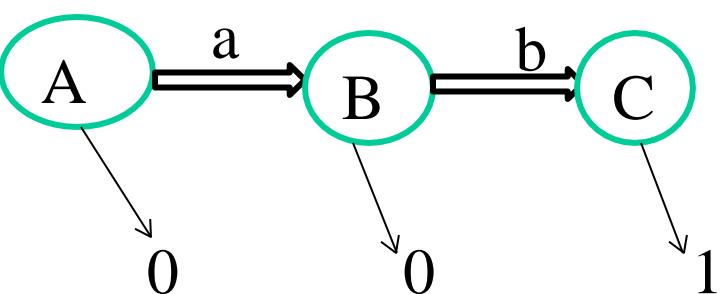
## Example on Moore Machine

Construct a Moore Machine that takes the set of all strings over  $\{a, b\}$  as i/p and prints '1' as o/p for every occurrence of 'ab' as a substring

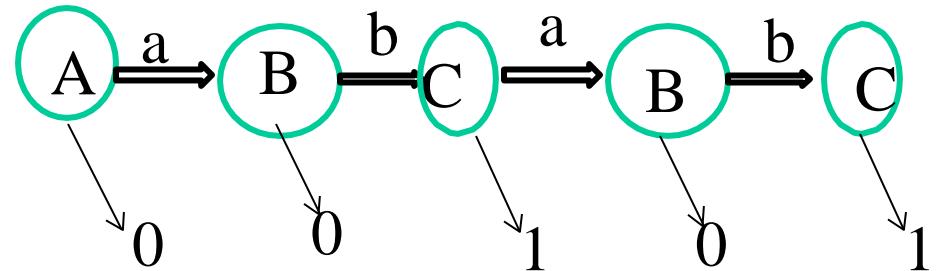
$$Q = \{A, B, C\} \quad \Sigma = \{a, b\} \quad \Delta = \{0, 1\}$$



• string 'ab' on machine



• string 'abab' on machine



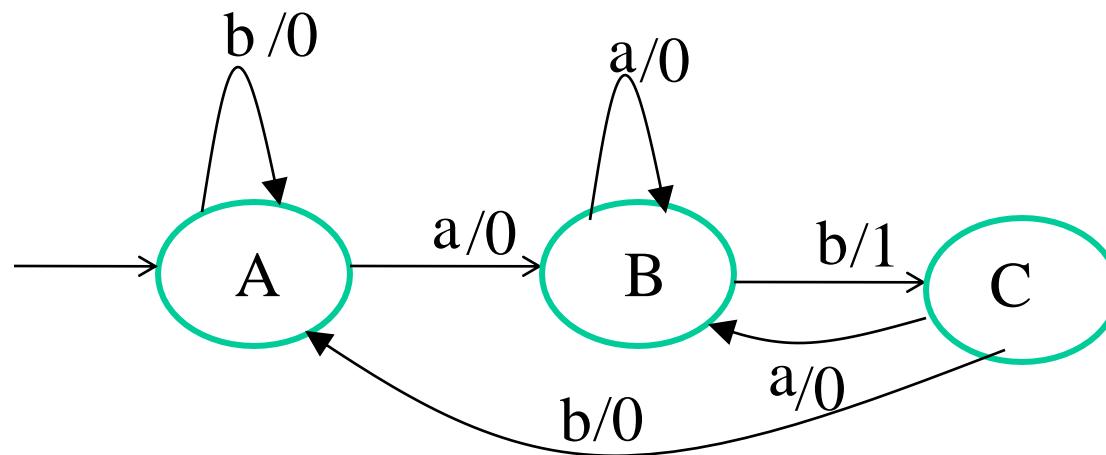
# Recall

- FA With O/P
- Moore Machine
- Mealy Machine

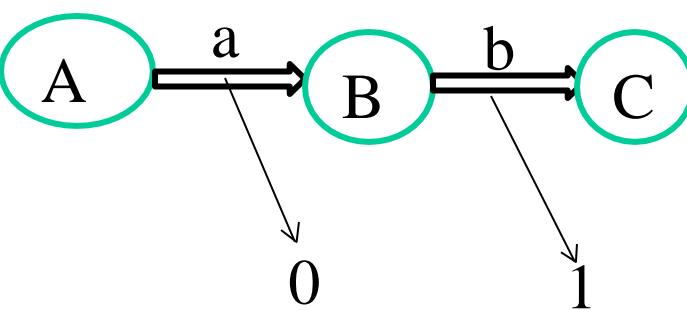
## Example on Mealy Machine

Construct a Mealy Machine that takes the set of all strings over  $\{a, b\}$  as i/p and prints '1' as o/p for every occurrence of 'ab' as a substring

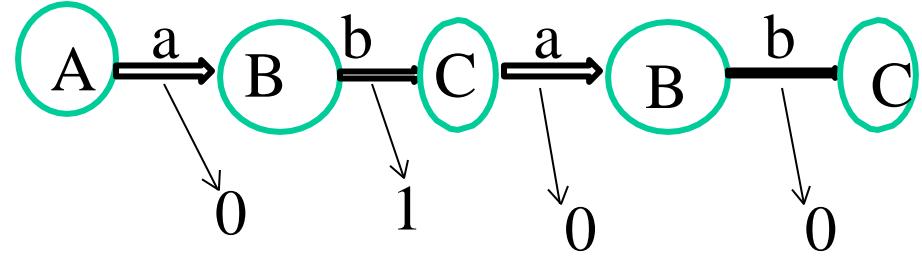
$$Q = \{A, B, C\}, \Sigma = \{a, b\}, \Delta = \{0, 1\}$$



• string 'ab' on machine

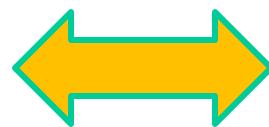


• string 'abab' on machine



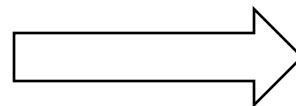
# Conversion of Moore and Mealy Machine

Moore Machine



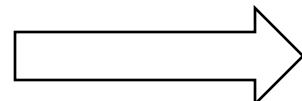
Mealy Machine

Moore Machine



Mealy Machine

Mealy Machine



Moore Machine

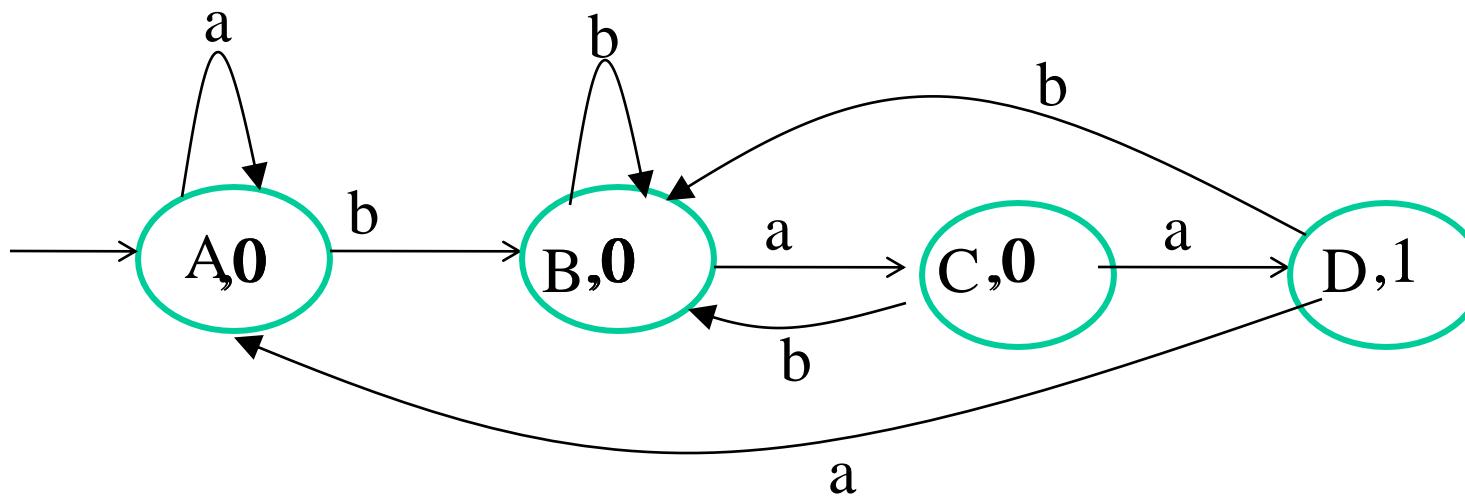
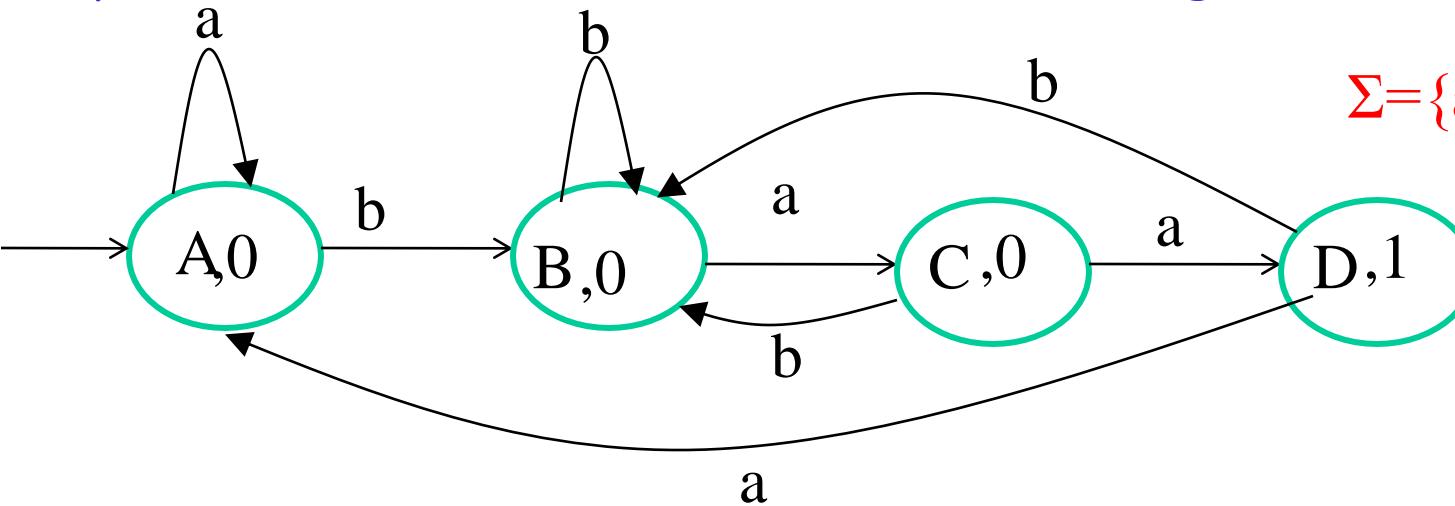
# Moore Machine to Mealy Machine

Construct a Moore Machine that takes the set of all strings over  $\{a, b\}$  as i/p and counts no of occurrences of substring 'baa'

$$Q = \{A, B, C\}$$

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

$$\Delta = \{0, 1\}$$



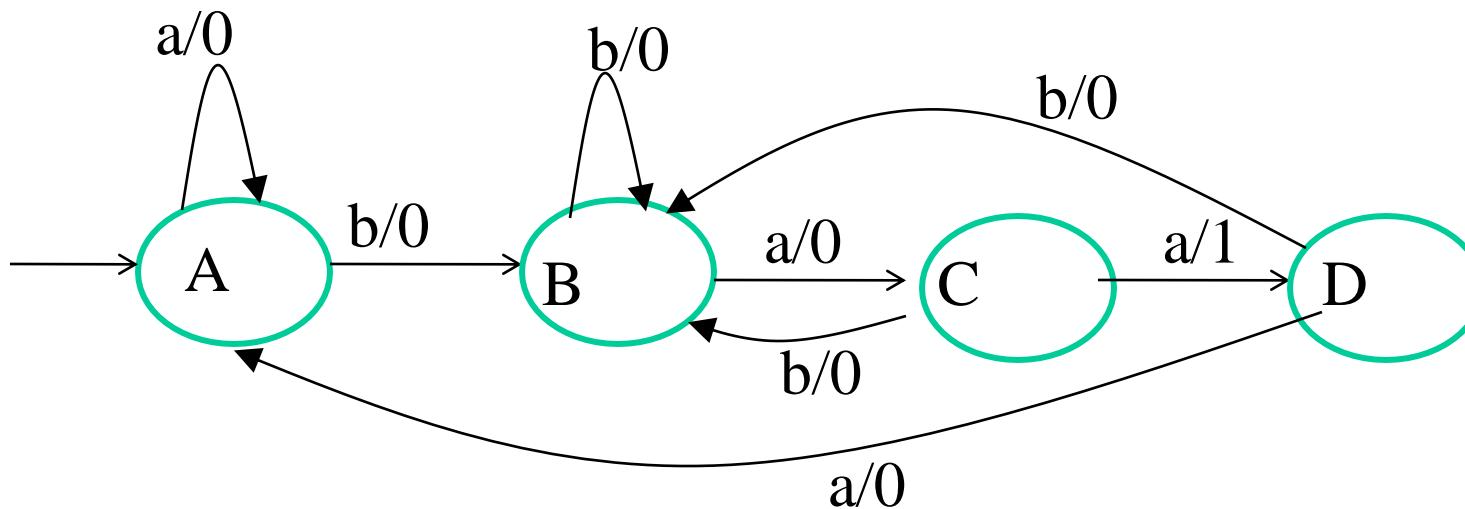
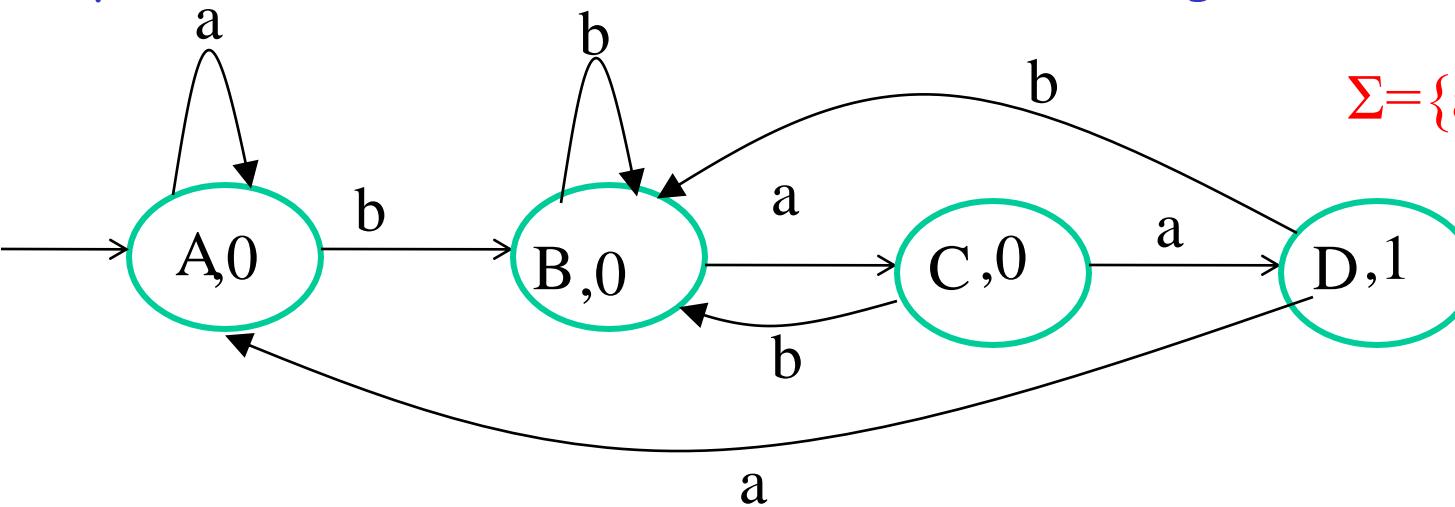
# Moore Machine to Mealy Machine

Construct a Moore Machine that takes the set of all strings over  $\{a, b\}$  as i/p and counts no of occurrences of substring 'baa'

$$Q = \{A, B, C\}$$

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

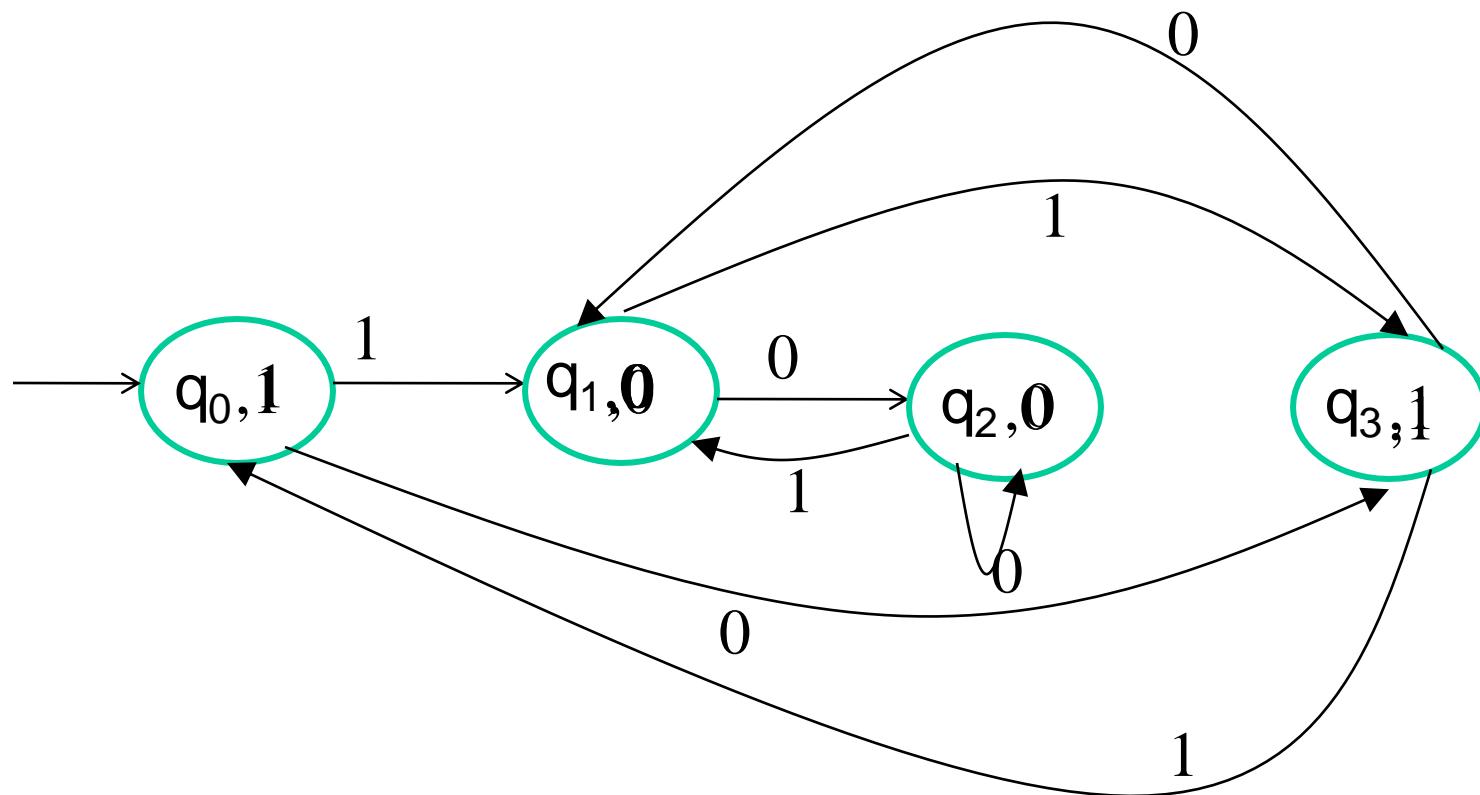
$$\Delta = \{0, 1\}$$



# Convert Following Moore Machine into Mealy Machine

Present State	Next State		Output
	$a = 0$	$a = 1$	
$\rightarrow q_0$	$q_3$	$q_1$	1
$q_1$	$q_2$	$q_3$	0
$q_2$	$q_2$	$q_1$	0
$q_3$	$q_1$	$q_0$	1

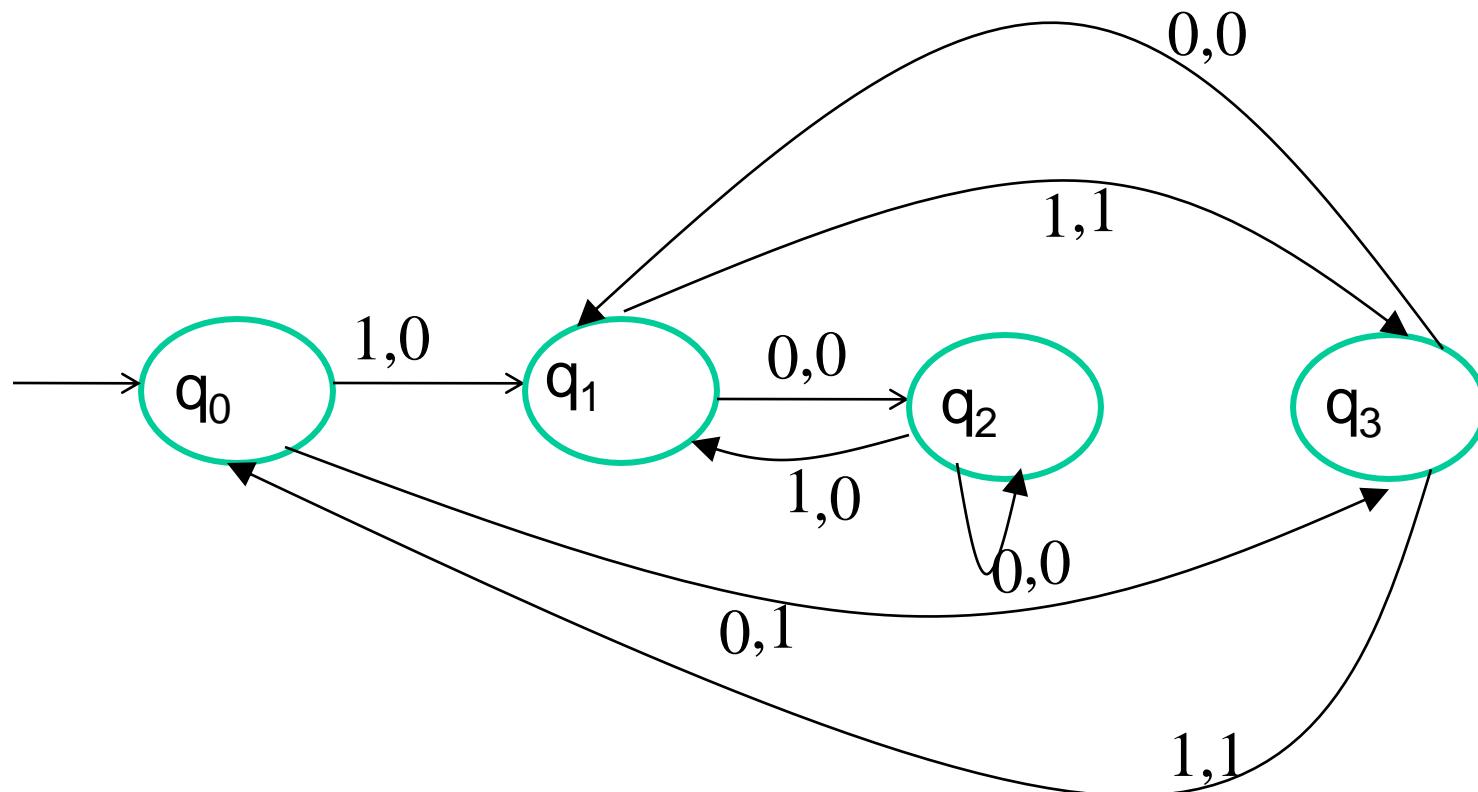
Aug-2015  
INSEM



# Convert Following Moore Machine into Mealy Machine

Present State	Next State		Output
	$a = 0$	$a = 1$	
$\rightarrow q_0$	$q_3$	$q_1$	1
$q_1$	$q_2$	$q_3$	0
$q_2$	$q_2$	$q_1$	0
$q_3$	$q_1$	$q_0$	1

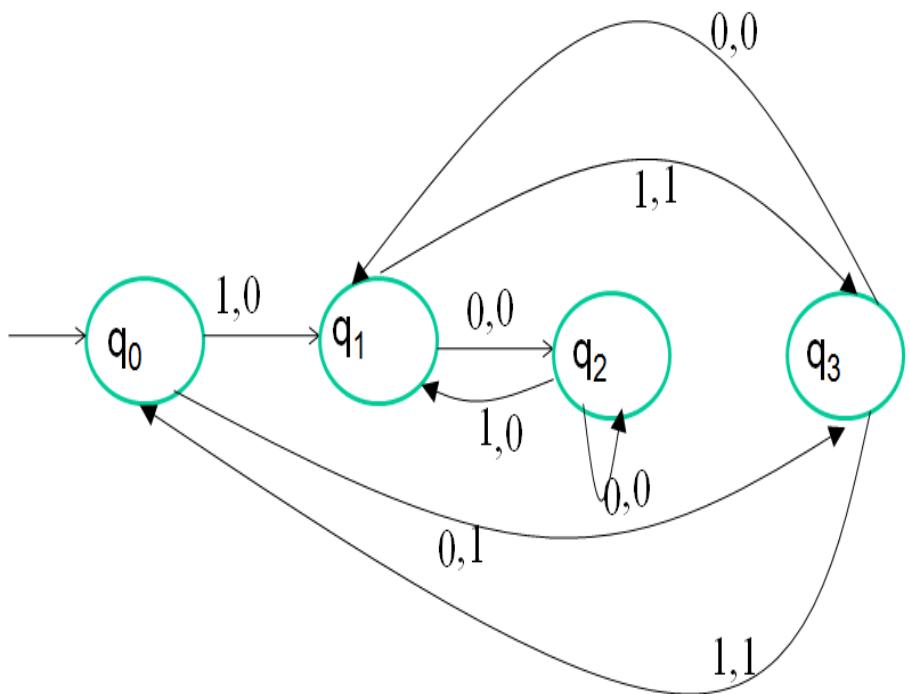
Aug-2015  
INSEM



# Convert Following Moore Machine into Mealy Machine

Present State	Next State		Output
	$a = 0$	$a = 1$	
$\rightarrow q_0$	$q_3$	$q_1$	1
$q_1$	$q_2$	$q_3$	0
$q_2$	$q_2$	$q_1$	0
$q_3$	$q_1$	$q_0$	1

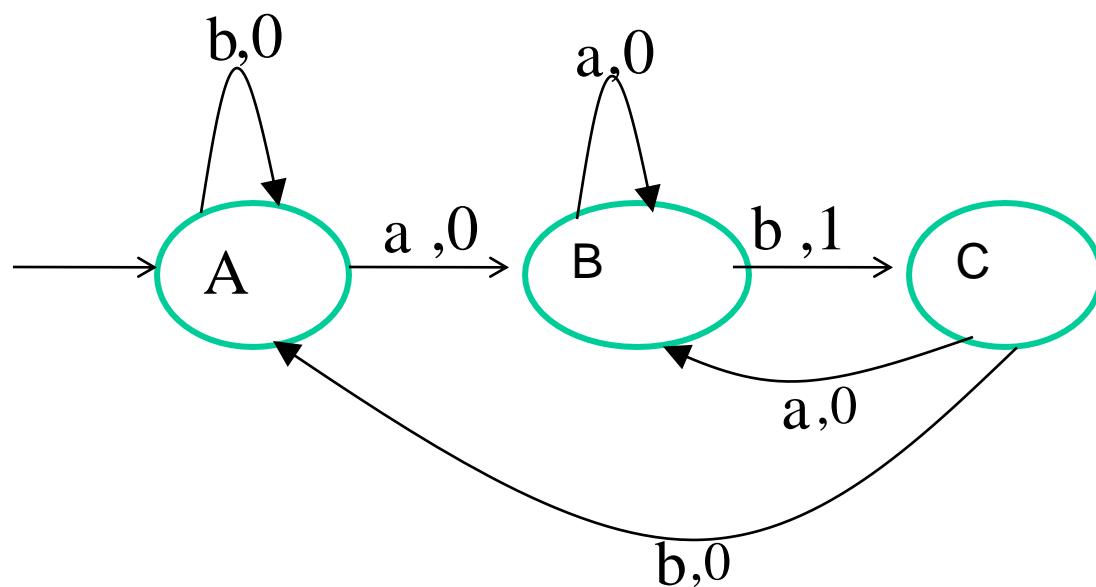
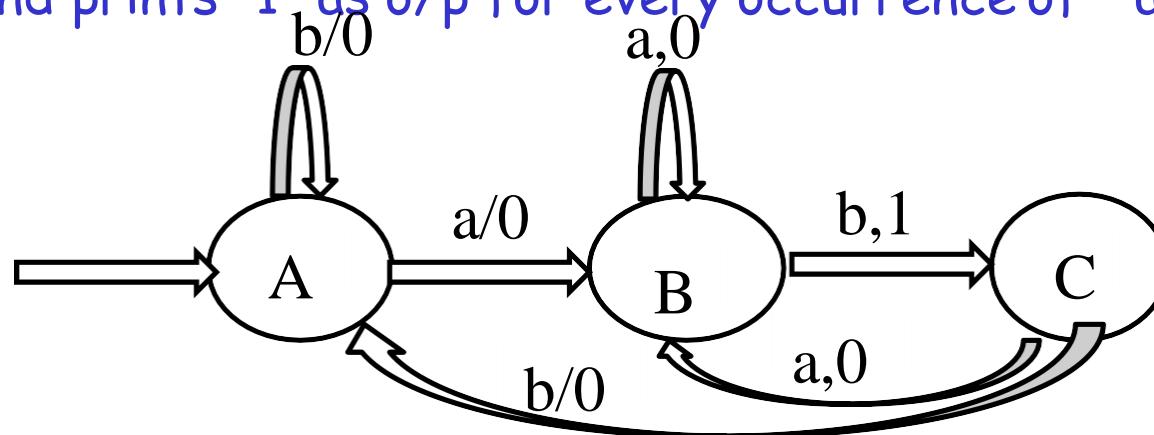
Aug-2015  
INSEM



	Input	Output	Input	Output
	0		1	
$q_0$	$q_3$	1	$q_1$	0
$q_1$	$q_2$	0	$q_3$	1
$q_2$	$q_2$	0	$q_1$	0
$q_3$	$q_1$	0	$q_0$	1

# Mealy Machine to Moory Machine

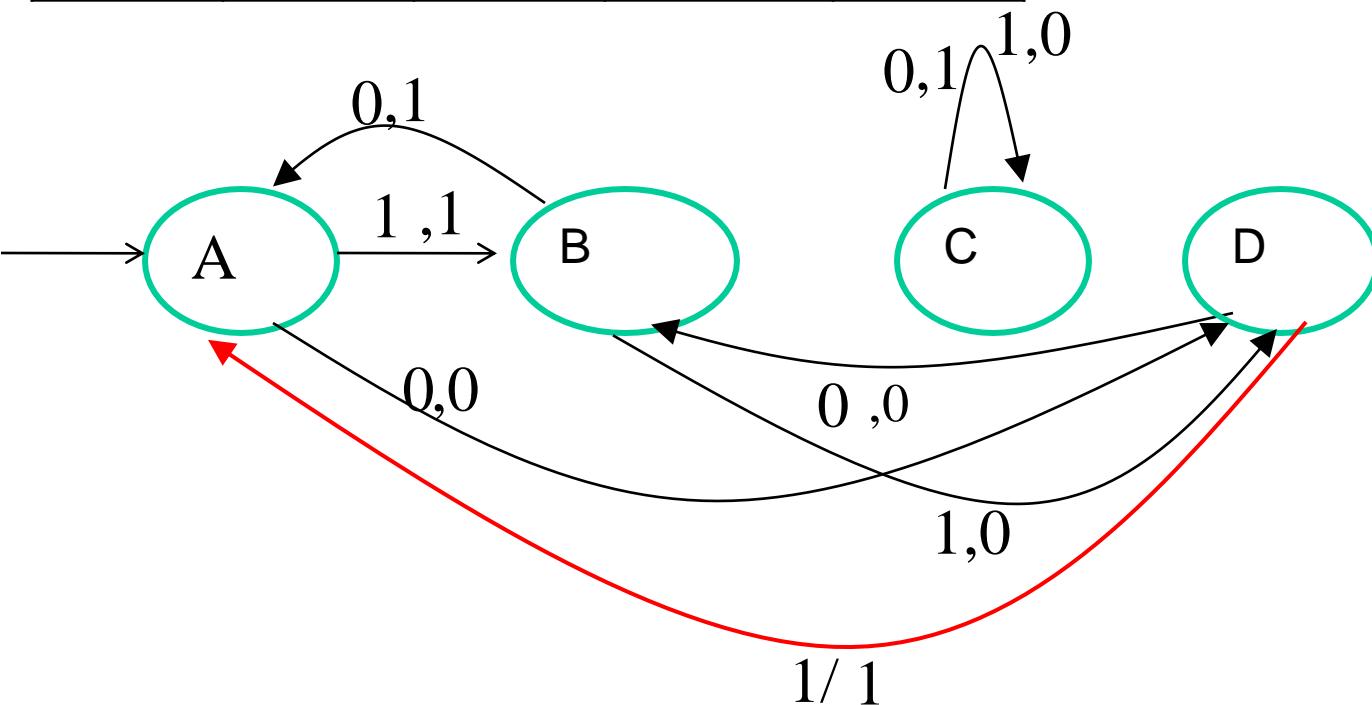
Construct a Mealy Machine that takes the set of all strings over  $\{a, b\}$  as i/p and prints '1' as o/p for every occurrence of 'ab' as a substring

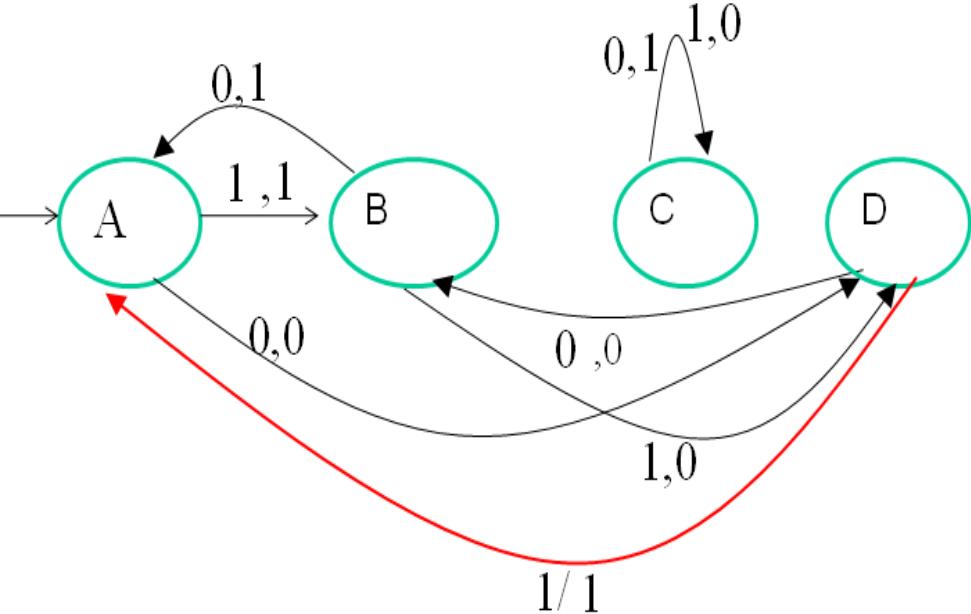


# Convert Following Mealy Machine into Moore Machine

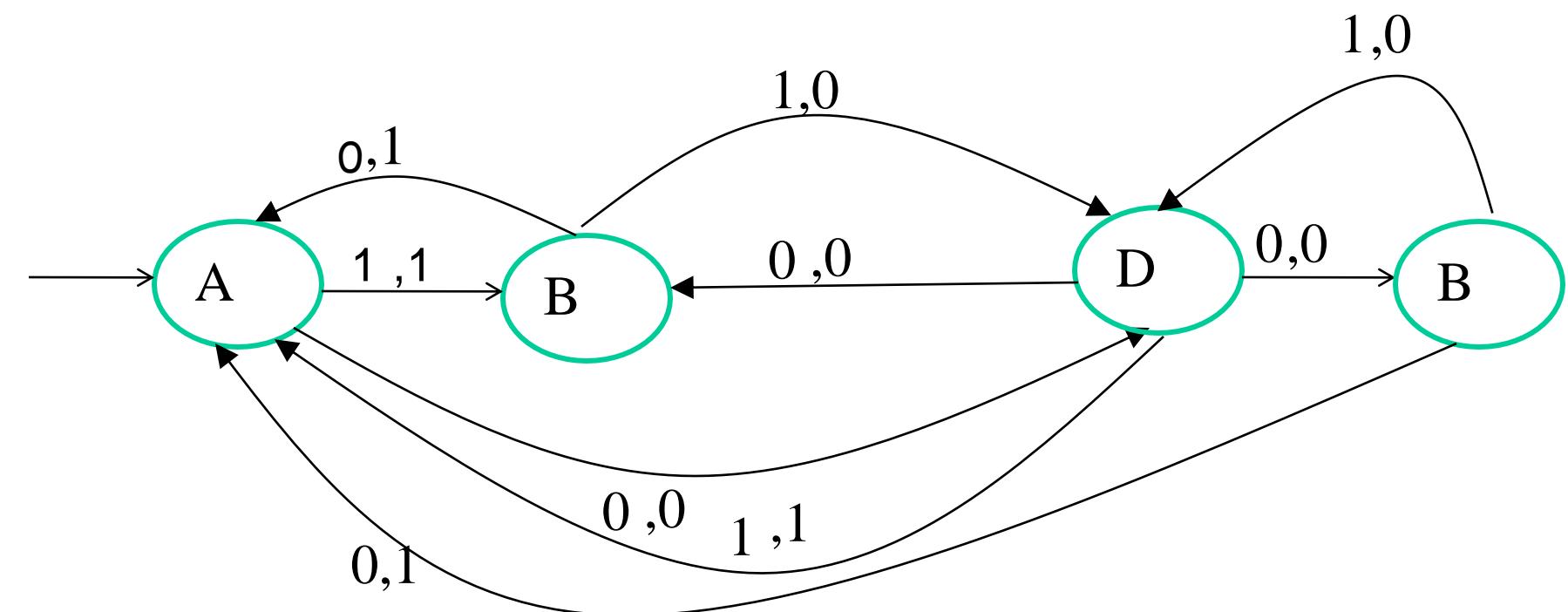
Present State	0		1	
	Next State	Output	Next State	Output
→ A	D	0	B	1
B	A	1	D	0
C	C	1	C	0
D	B	0	A	1

Aug-2015  
INSEM





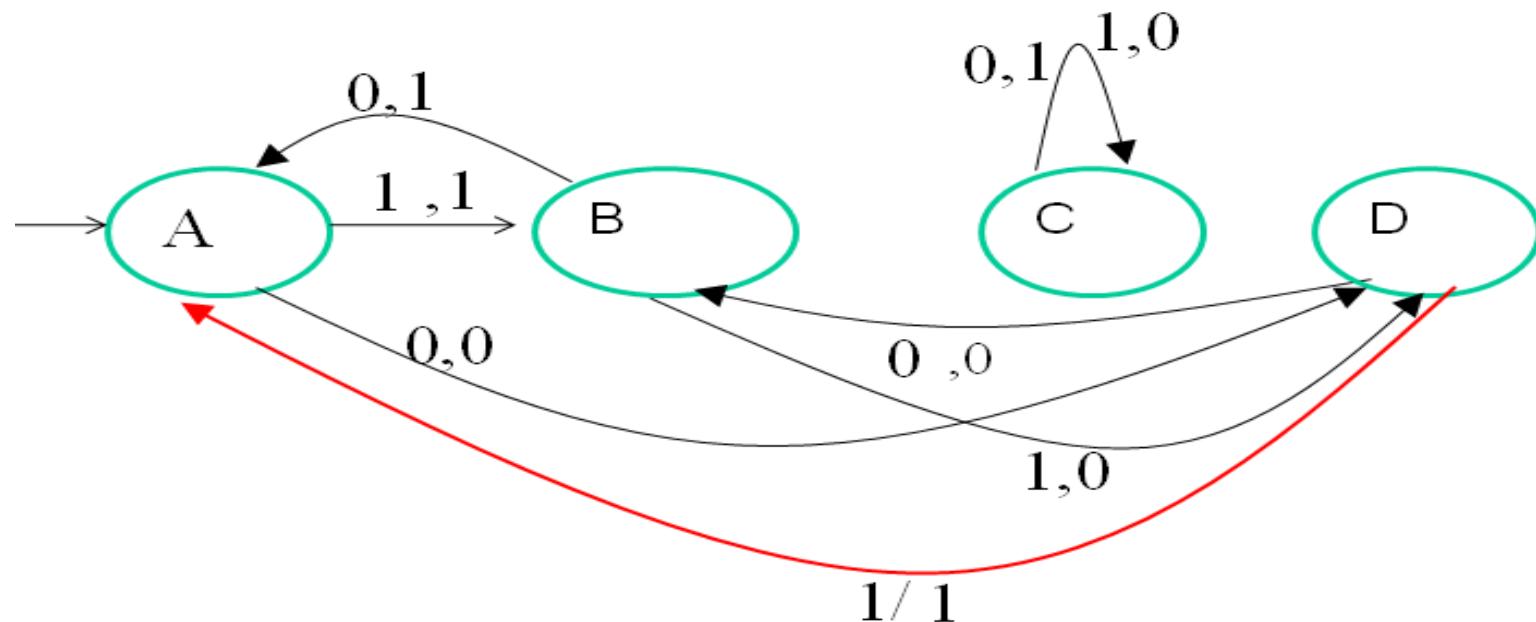
Present State	0		1	
	Next State	Output	Next State	Output
$\rightarrow A$	D	0	B	1
B	A	1	D	0
C	C	1	C	0
D	B	0	A	1



# Convert Following Mealy Machine into Moore Machine

Present State	0		1	
	Next State	Output	Next State	Output
→ A	D	0	B	1
B	A	1	D	0
C	C	1	C	0
D	B	0	A	1

Aug-2015  
INSEM



Present State	0		1	
	Next State	Output	Next State	Output
→ A	D	0	B	1
B	A	1	D	0
C	C	1	C	0
D	B	0	A	1

Present State	Input		Output
	0	1	
A	D	B1	1
D	B0	A	0
B0	A	D	0
B1	A	D	1
C0	C1	C0	0
C1	C1	C0	1

b) Construct Mealy machine equivalent to the given Moore machine

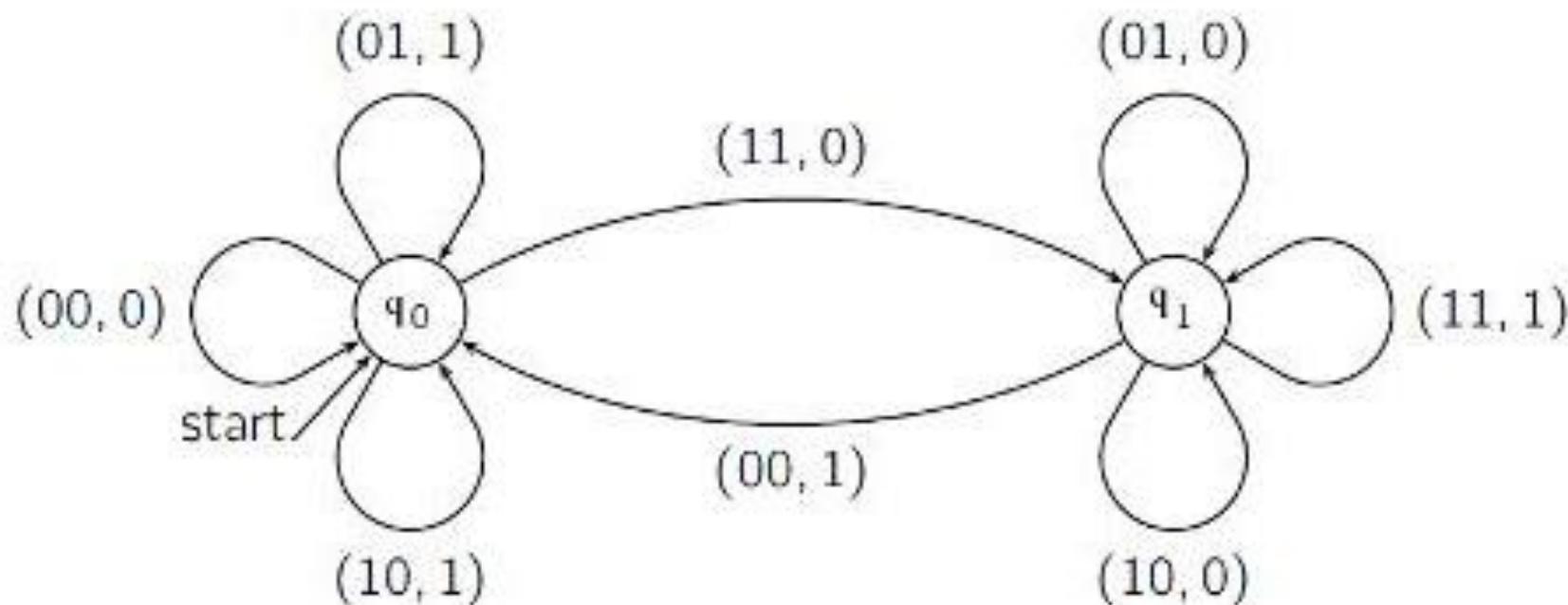
Aug-2017

INSEM

	0	1	O/P
q0	q0	q1	N
q1	q0	q2	N
q2	q0	q3	N
q3	q0	q3	Y

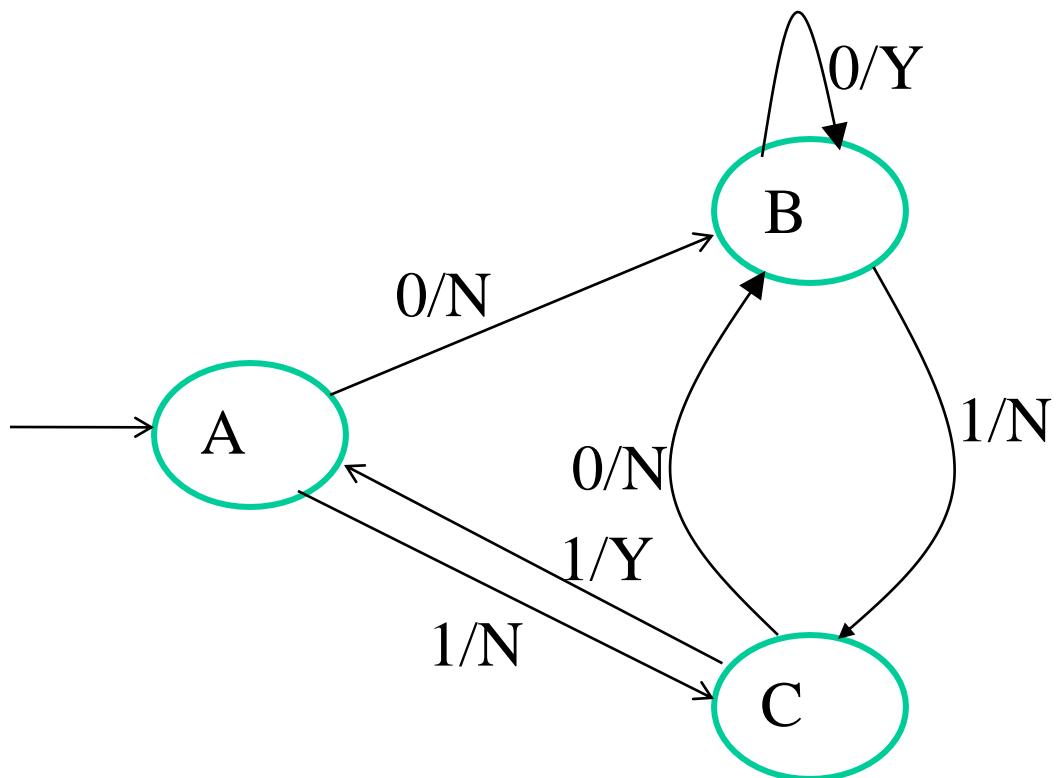
Start state : q0 ; Final state : q3

Design a finite automata which perform addition of two Binary number.



Design Moore Machine for divisibility by 3  
tester for binary number. (Nov-2017 6 Marks)

Convert following Mealy Machine to Moore  
Machine (Nov-2017 6 Marks)



- University Question Solving Session

Thank you!!!!!!