COM3064 Automata Theory

Week 2: Deterministic Finite Automata

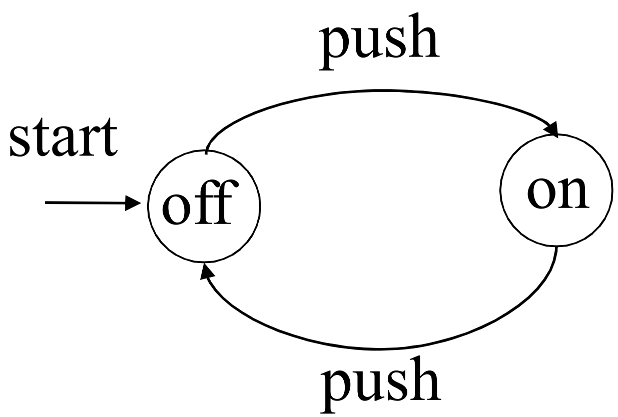
Lecturer: Dr. Sevgi YİĞİT SERT
Spring 2023

Resources: Introduction to The Theory of Computation, M. Sipser,
Introduction to Automata Theory, Languages, and Computation, J.E. Hopcroft, R. Motwani, and J.D. Ullman
BBM401 Automata Theory and Formal Languages, İlyas Çiçekli

Finite Automata

- A **Finite automata** has ***finite number of states*** connected by ***transition rules*** that take you from one state to another.
- The ***purpose of a state*** is to remember the relevant portion of the system's history.
  - Since there are only a ***finite number of states***, the entire history cannot be remembered.
    - So the system must be designed carefully to remember what is important and forget what is not.
  - The advantage of having only a finite number of states is that we can implement the system with a fixed set of resources.
    - a circuit or a simple form of program.

A Simple Finite Automaton – On/Off Switch

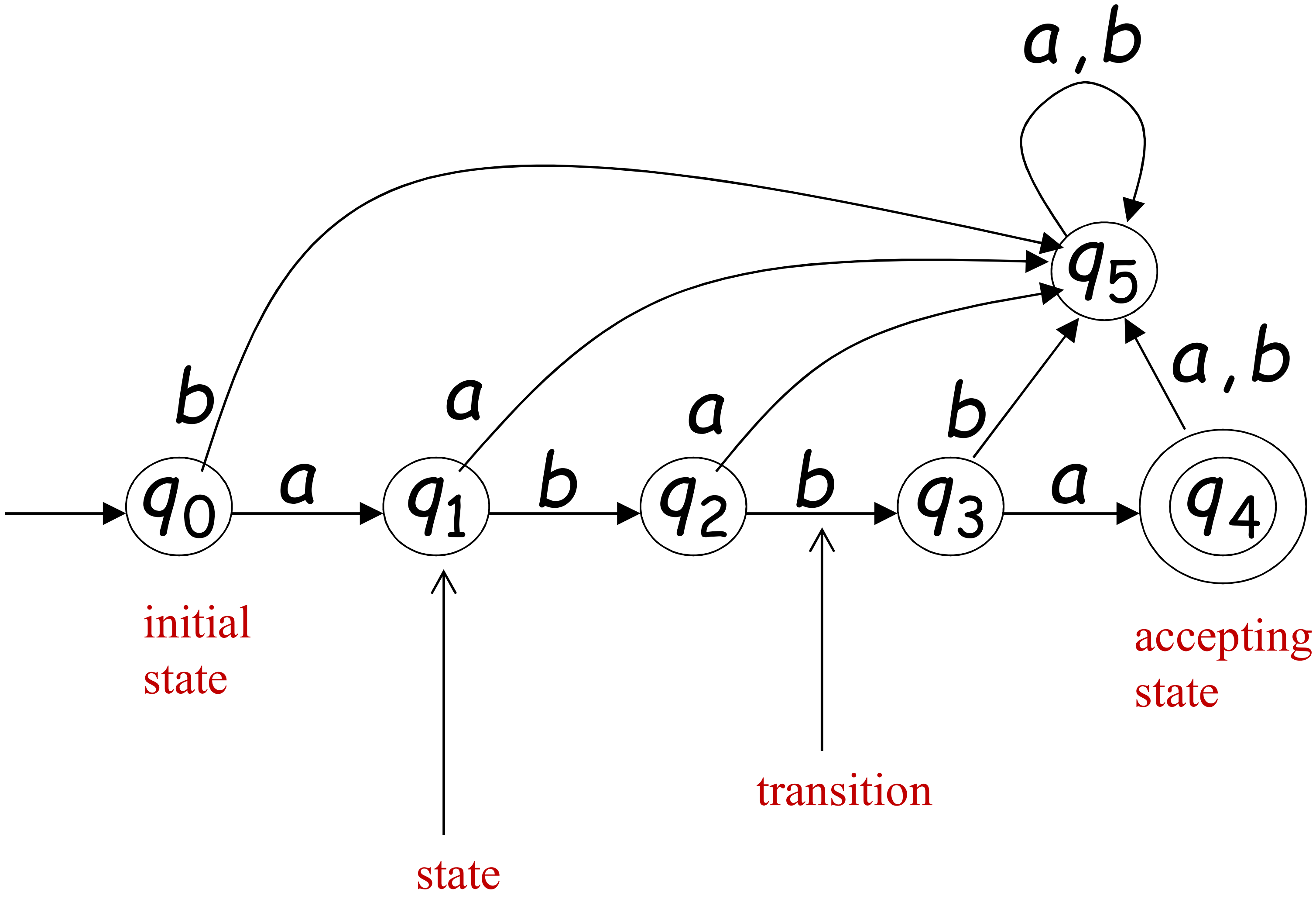


In a **finite automaton**:

- **States** are represented by **circles**.
- **Accepting (final) states** are represented by **double circles**.
- One of the states is a **starting state**.
- **Arcs** represent **state transitions** and **labels on arcs** represent **inputs** causing transitions.

- The on/off switch remembers whether it is in the on-state or the off-state.
  - It allows the user to press a button whose effect is different depending on the state of the switch.

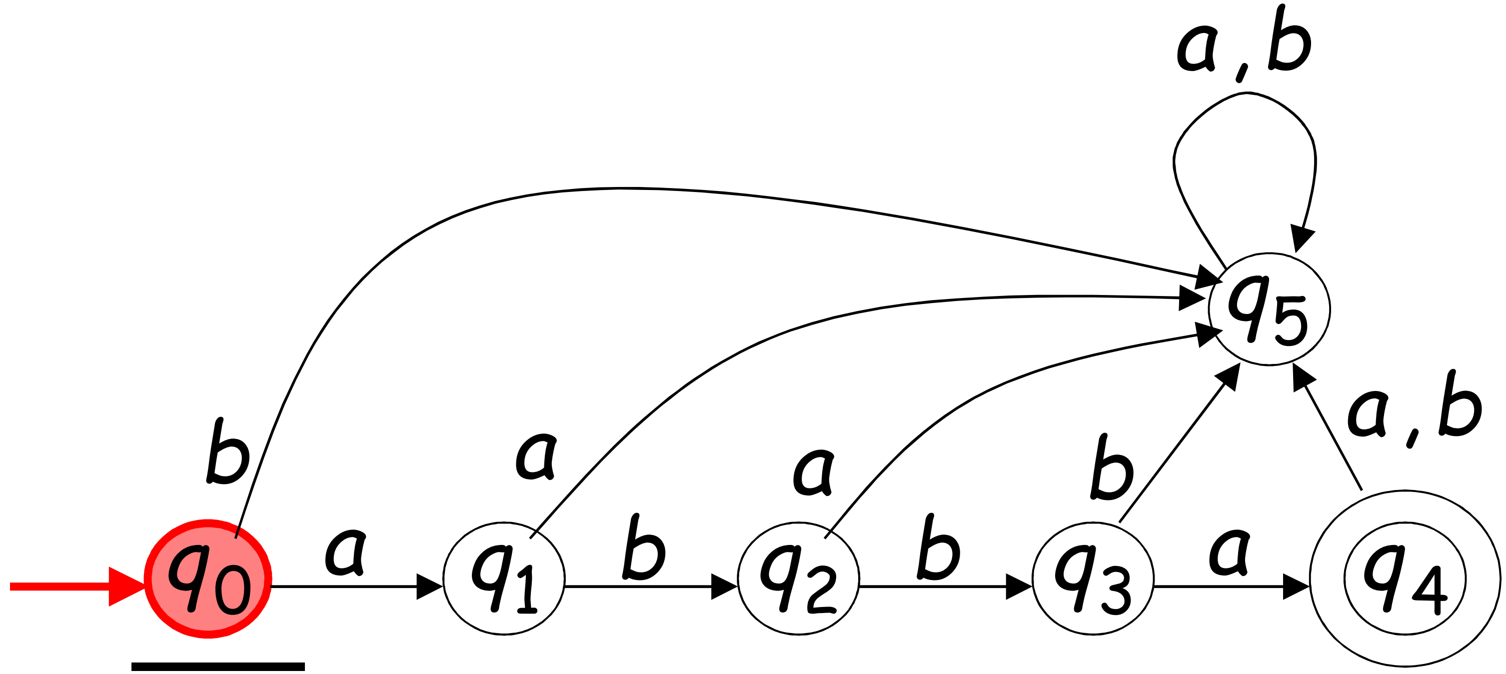
A Finite Automation



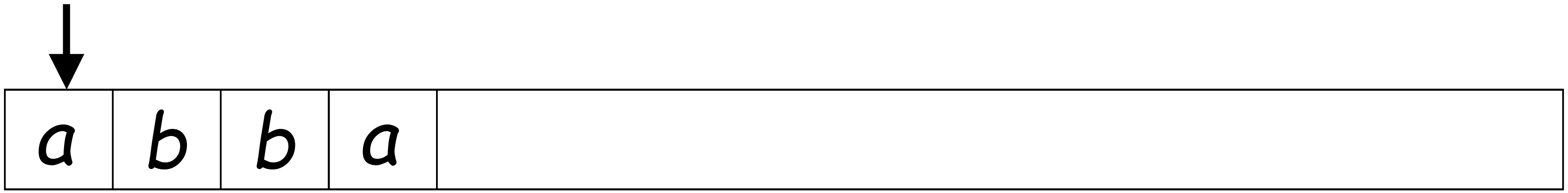
Initial Configuration

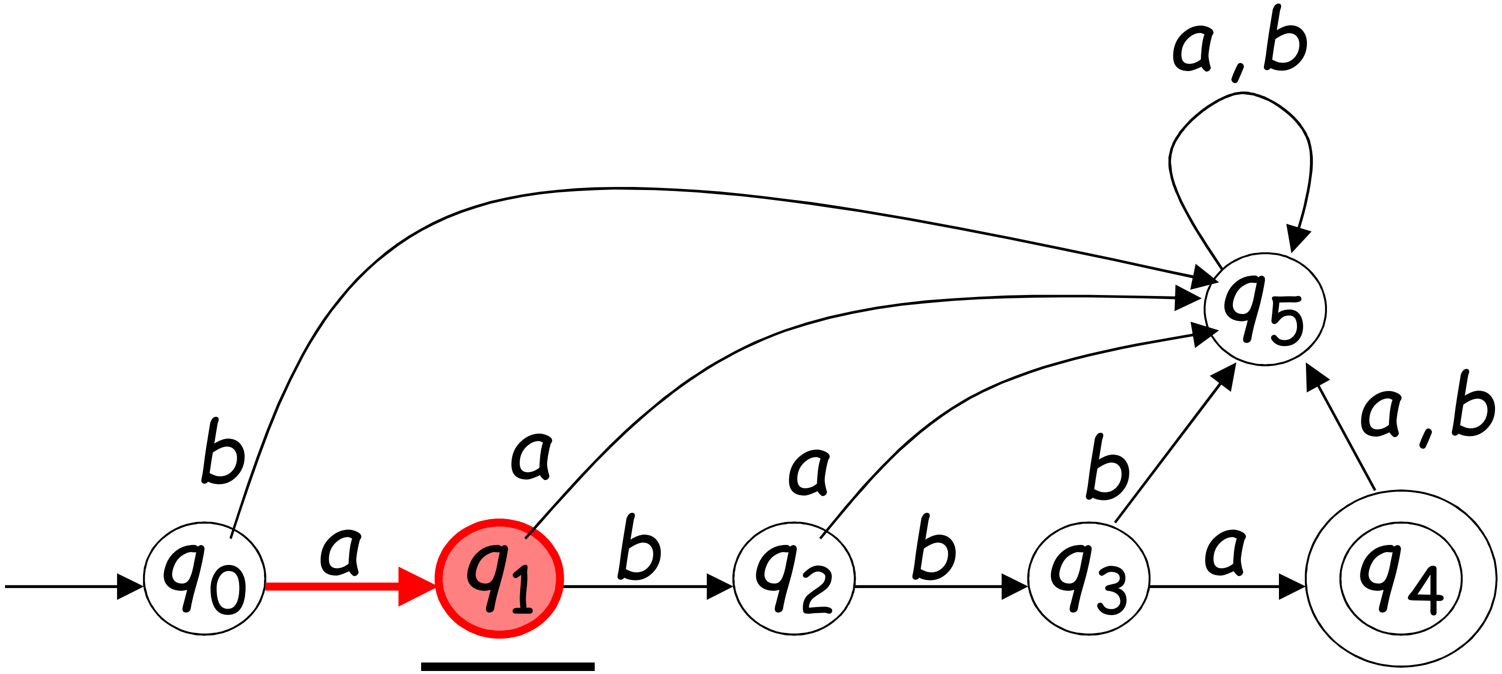
Input String

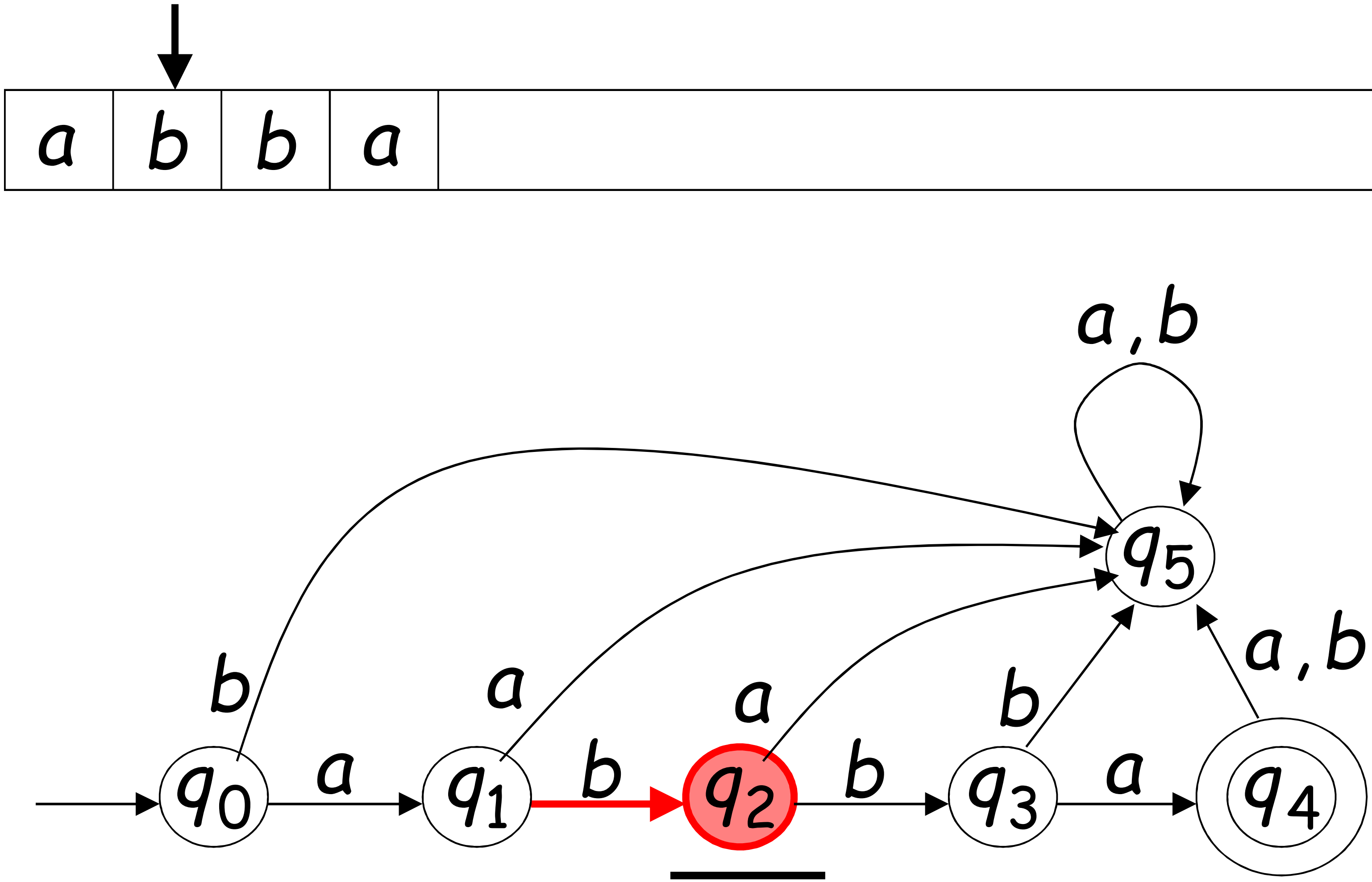
a	b	b	a	



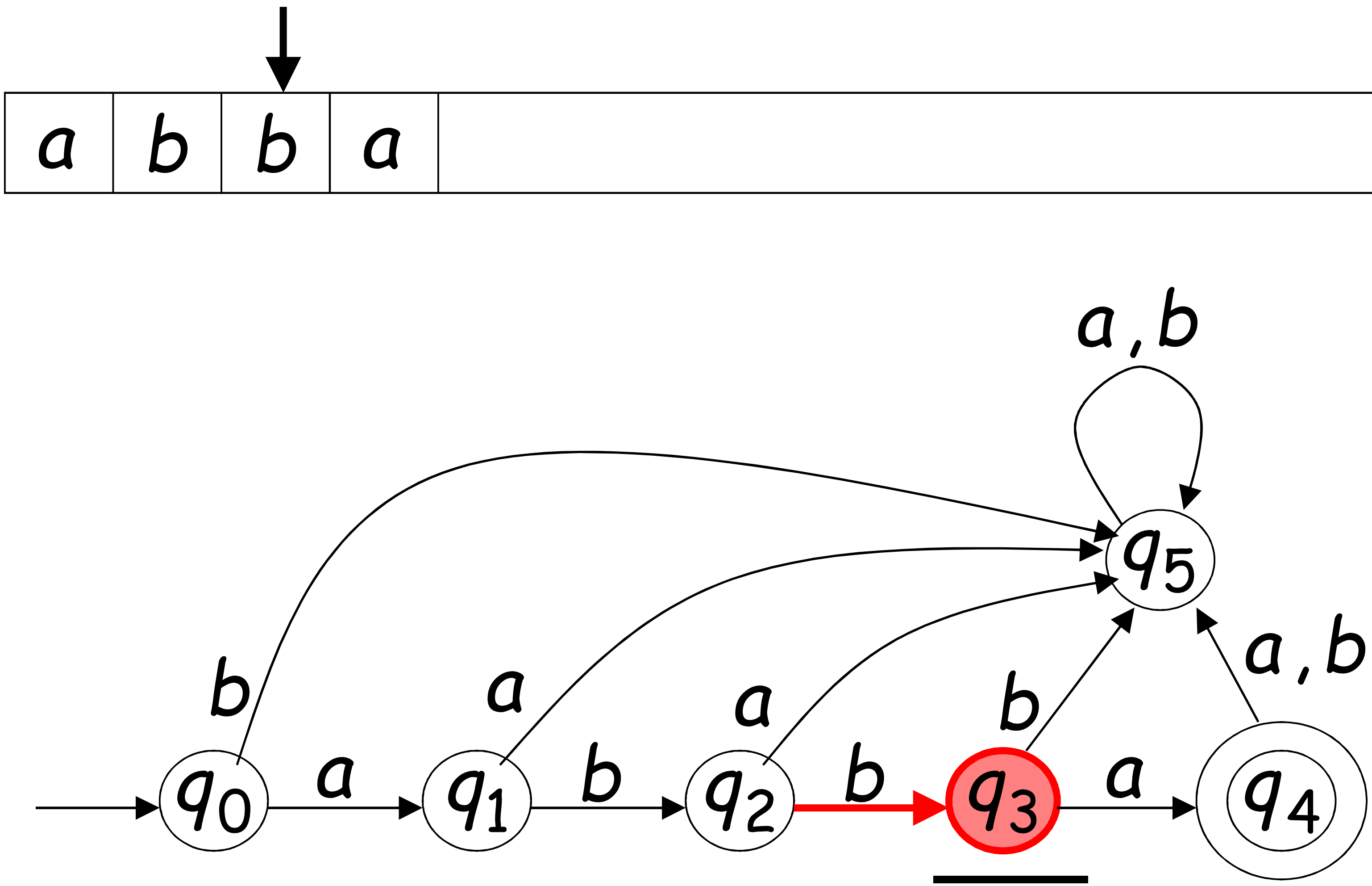
Reading the Input

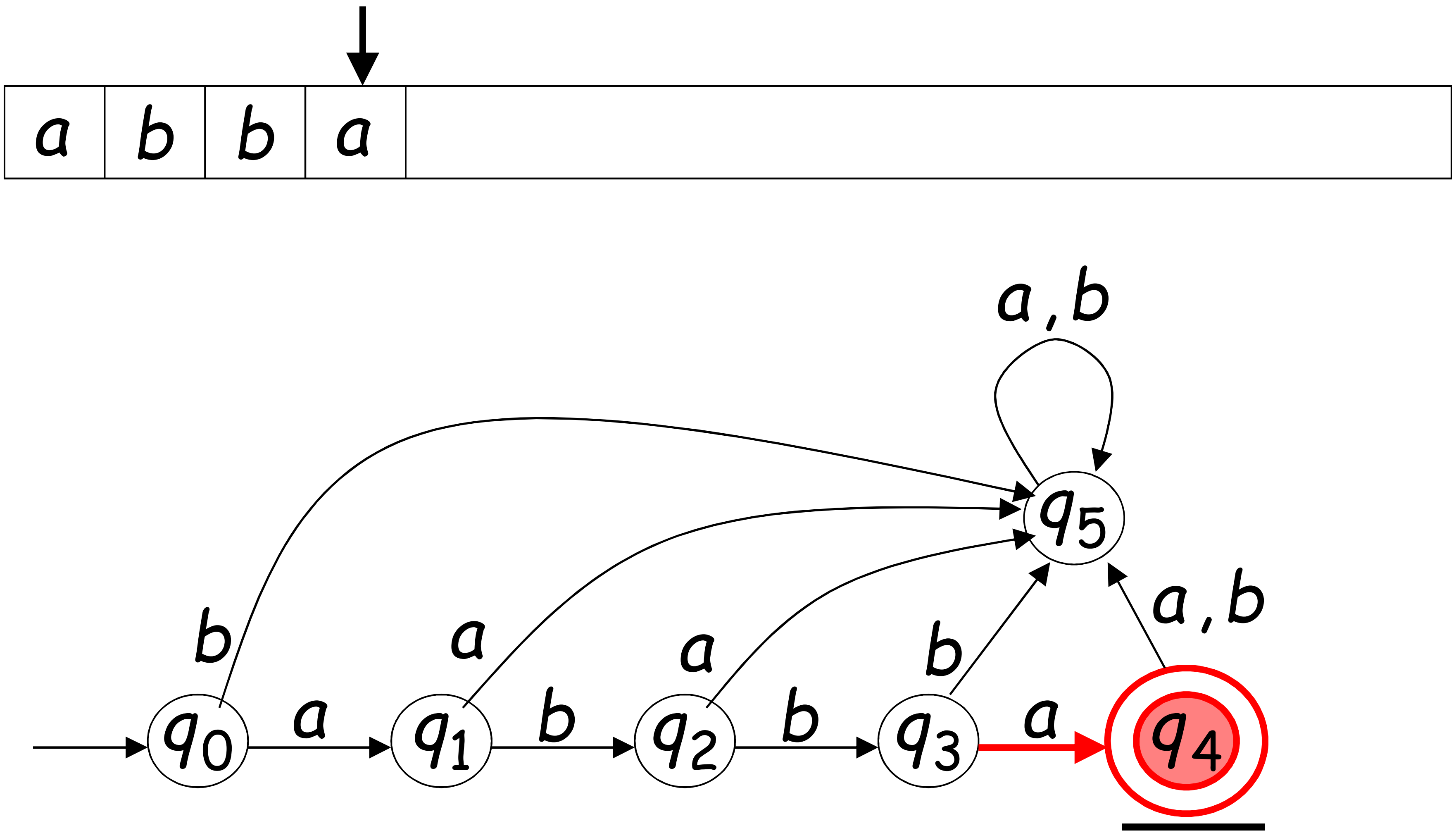






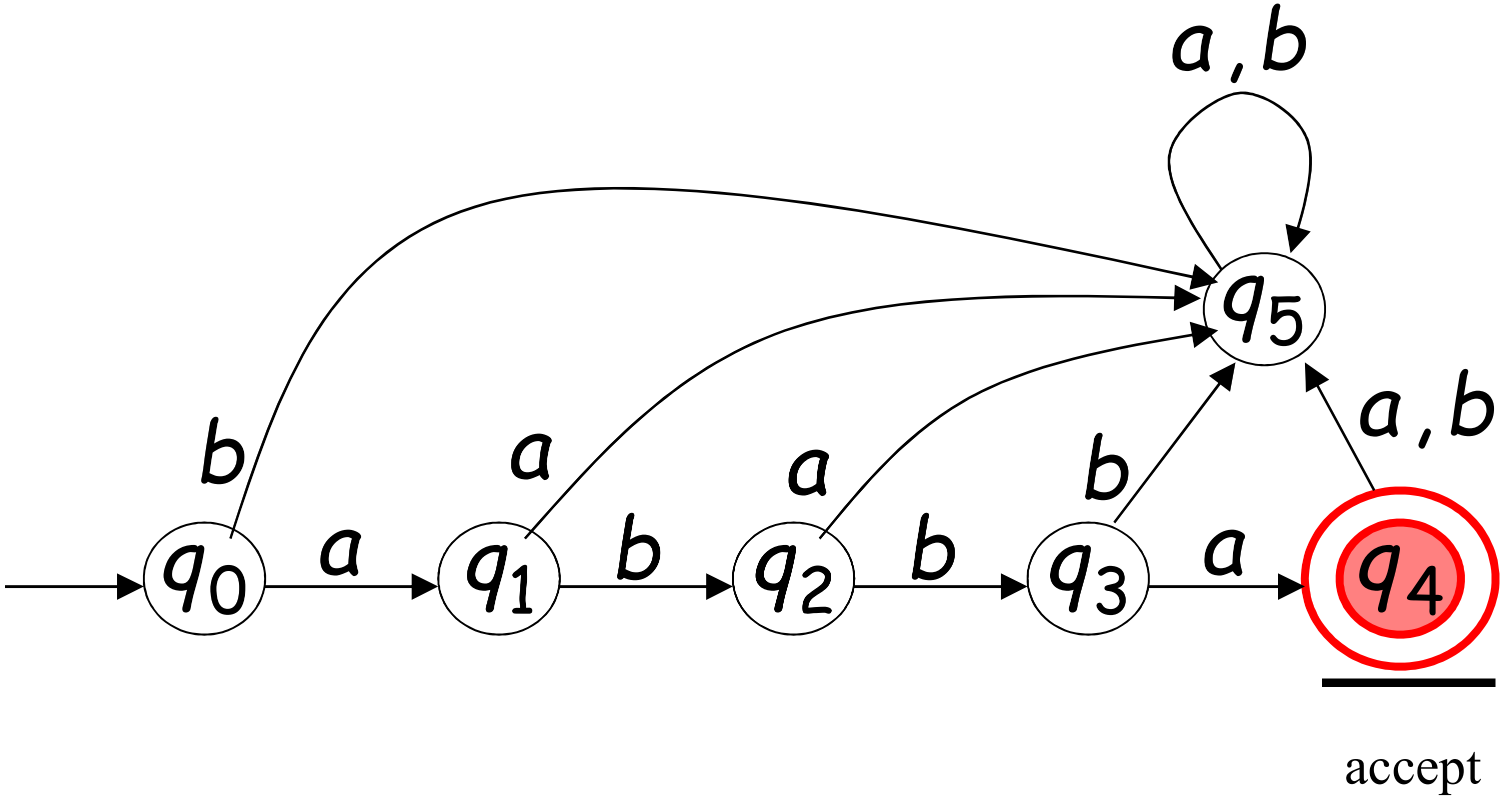
a b b a
a,b
b
a
a
b
a
b
a
a,b
q_0
q_1
q_2
q_3
q_4
q_5


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

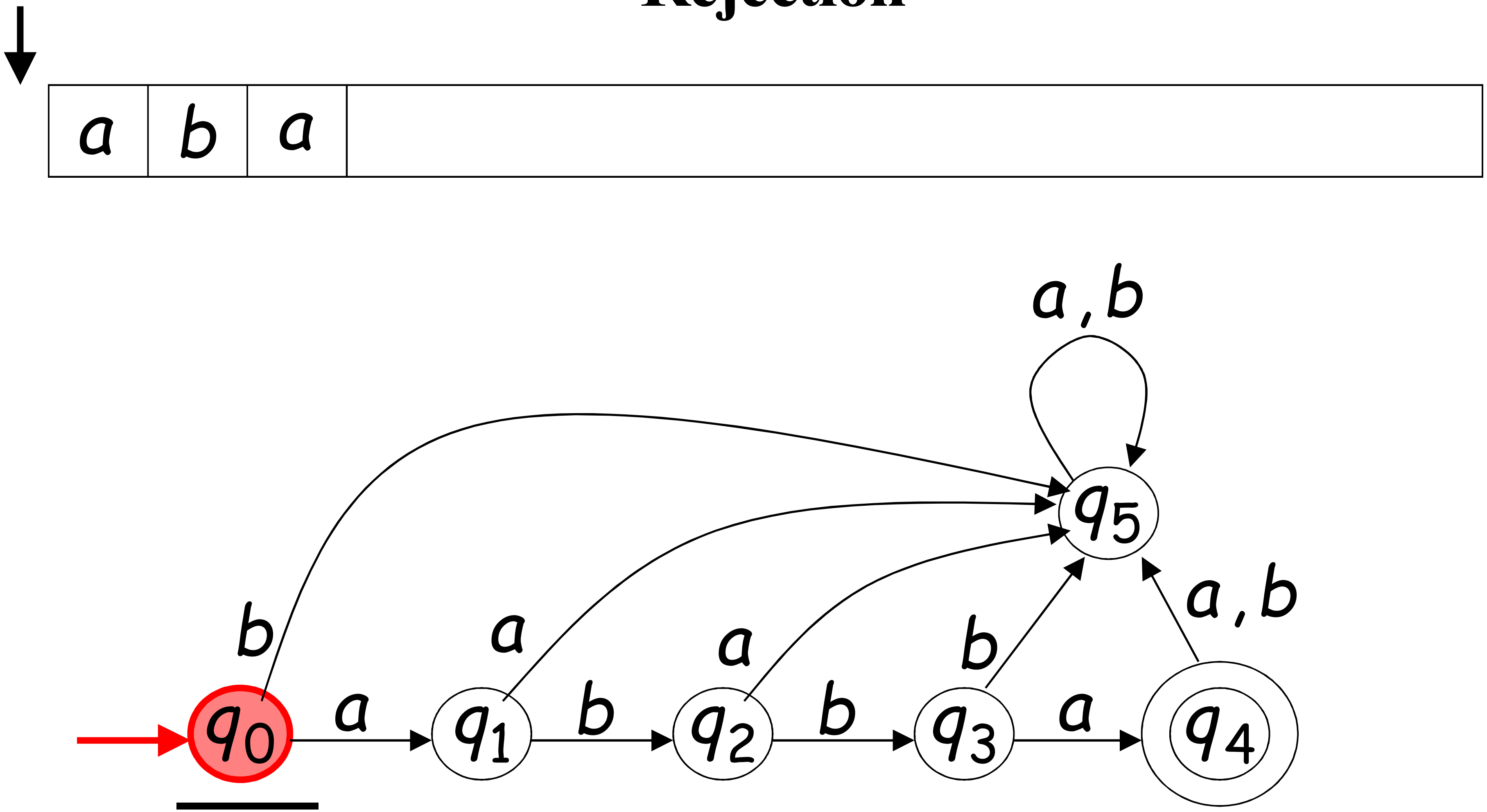

a
b
b
a
a, b
b
a
a
b
a
a, b
q_0
a
q_1
b
q_2
b
q_3
a
q_4
q_5

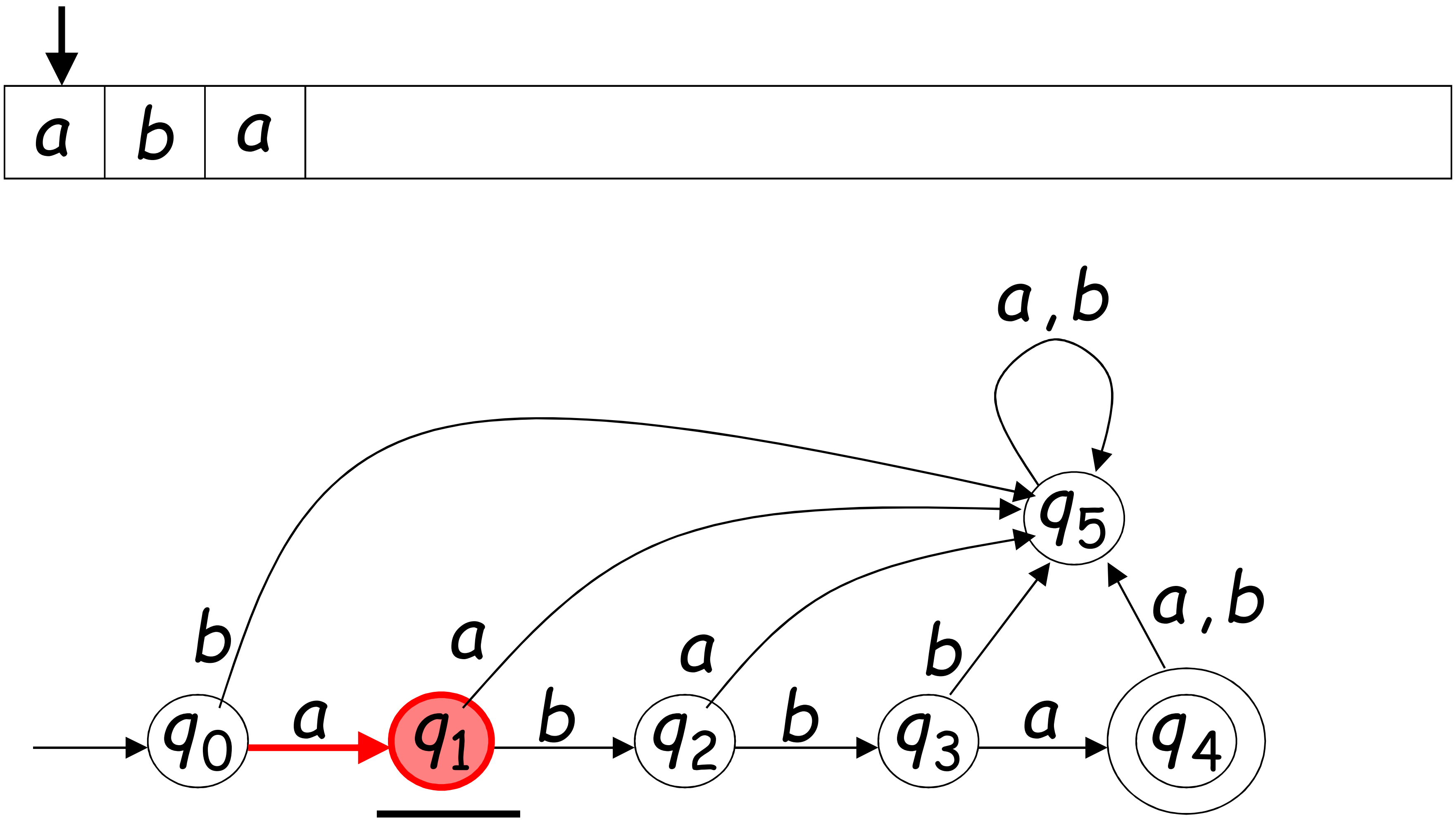
Input finished

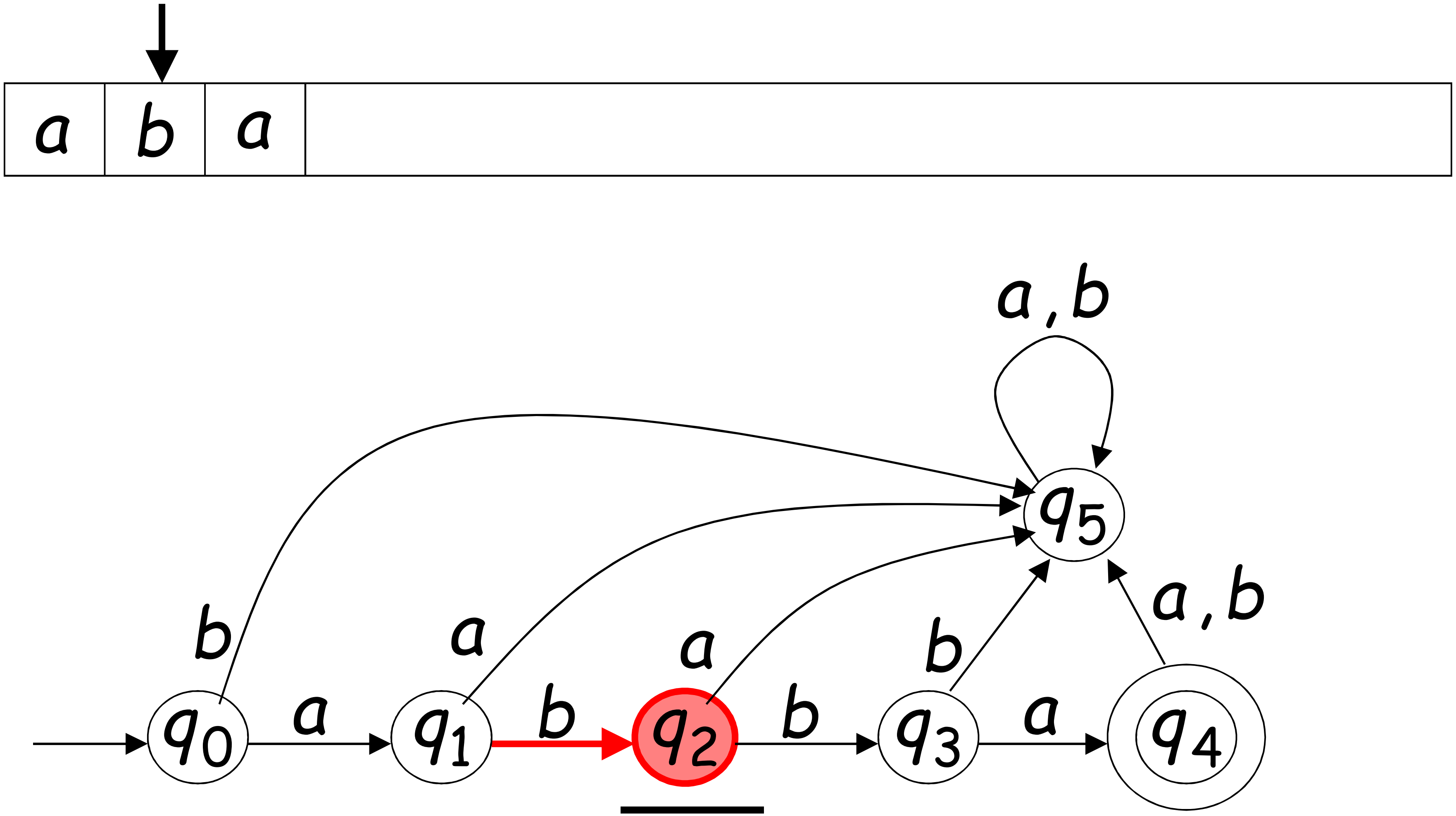
a	b	b	a	



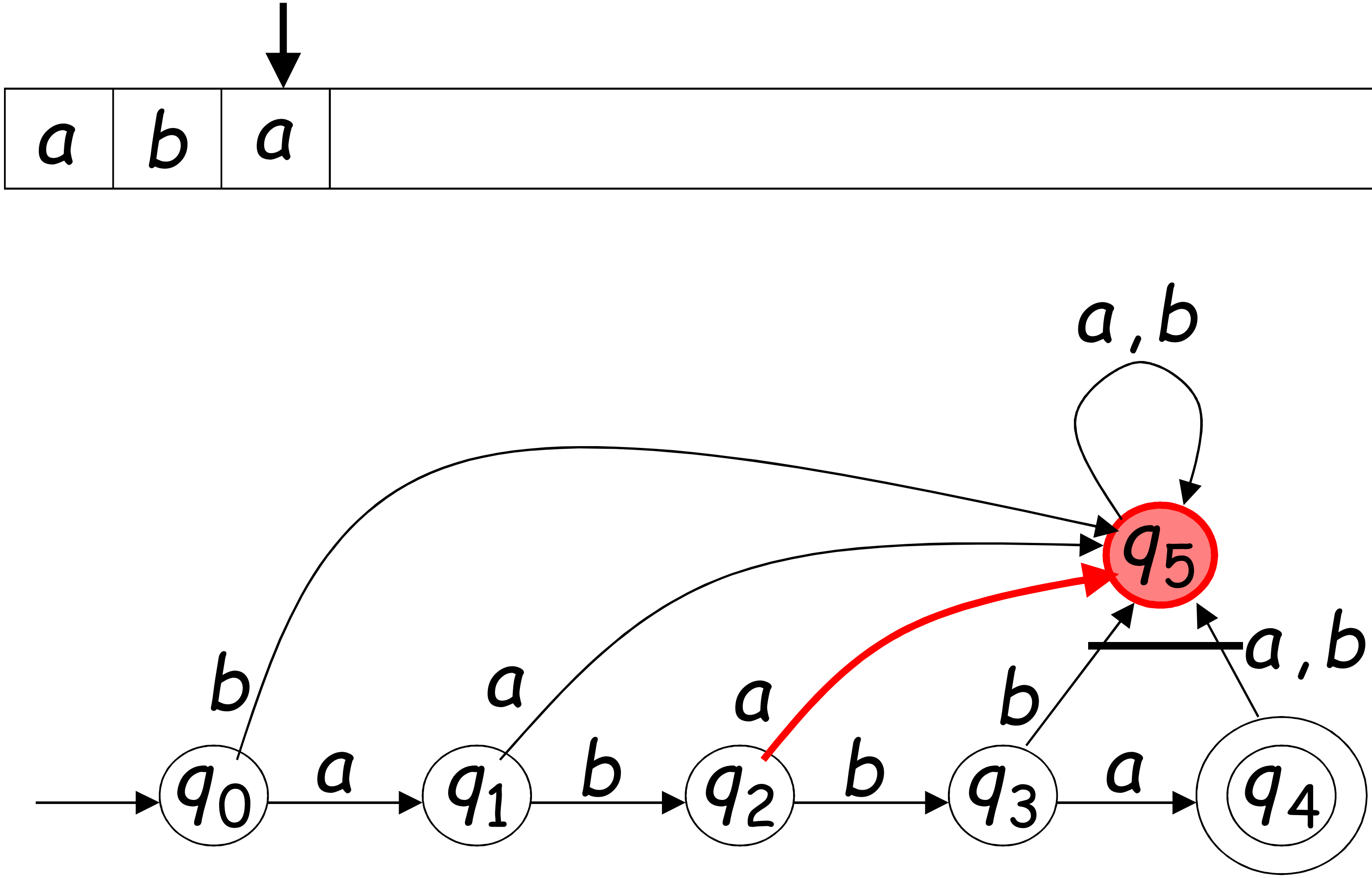
Rejection




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

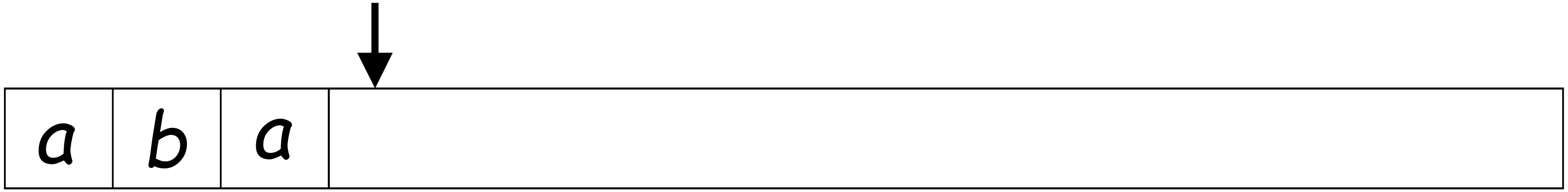


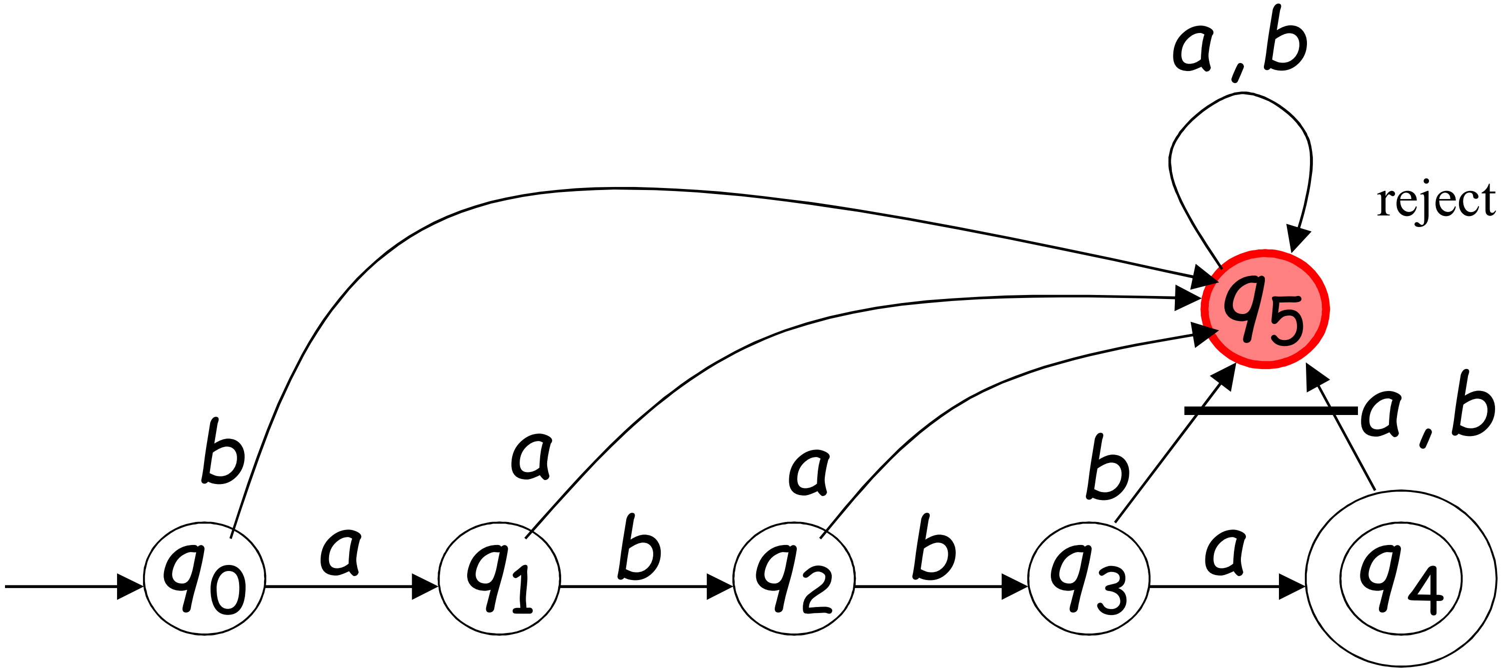
a
b
a
a,b
q_5
b
a
a
b
a,b
q_0
a
q_1
b
q_2
b
q_3
a
q_4



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

Input finished

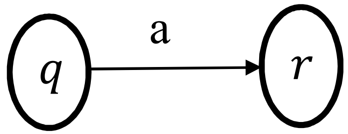




Deterministic Finite Automaton (DFA)

A **Deterministic Finite Automaton (DFA)** is a 5-tuple

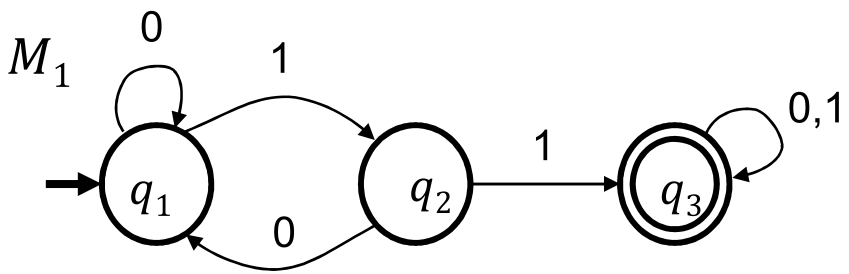
A = (Q, Σ, δ, q_0, F)

1. Q is a **finite set of states**
2. Σ is a **finite set of symbols** (alphabet)
3. Delta (δ) is a **transition function** $\delta\ (q,\ a)\ =\ r$ means 

4. q_0 is the **start state** ($q_0 \in$ Q)
5. F is a set of **final (accepting) states** (F $\subseteq$ Q)

- Transition function takes two arguments: a state and an input symbol.
- $\delta\ (q, a)\ =$ the state that the DFA goes to when it is in state q and input a is received.

Graph Representation of DFA



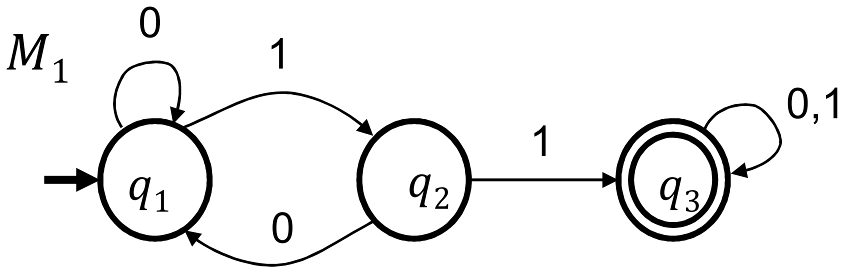
- Nodes = states.
- Arcs represent transition function.
  - Arc from state p to state q labeled by all those input symbols that have transitions from p to q.
- Arrow labeled "Start" to the start state.
- Final states indicated by double circles.

Graph Representation of DFA

A DFA: Accepts all strings contain substring 11

M_1 accepts exactly those strings in A where $A = \{w |\ w \text{ contains substring } 11\}$.

.



$M_1 = (Q, \Sigma, \delta, q_1, F)$

$Q = \{q_1, q_2, q_3\}$ $\Sigma = \{0, 1\}$ $F = \{q_3\}$

- States:
  - State q_1: previous string is NOT OKAY (does not contain 11), and it contains none of 1s.
  - State q_2: previous string is NOT OKAY (does not contain 11), and it contains a single 1.
  - State q_3: previous string contains two consecutive 1's (it is OKAY).

Say that A is the language of M_1 and that M_1 recognizes A and that $A = L(M_1)$.

Alternative Representation: Transition Table

- $\delta\,(q_1,\ 0) = \ q_1$
- $\delta\,(q_1,\ 1) = \ q_2$
- $\delta\,(q_2,\ 0) = \ q_1$
- $\delta\,(q_2,\ 1) = \ q_3$
- $\delta\,(q_3,\ 0) = \ q_3$
- $\delta\,(q_3,\ 1) = \ q_3$

Columns = input symbols

$\delta =$

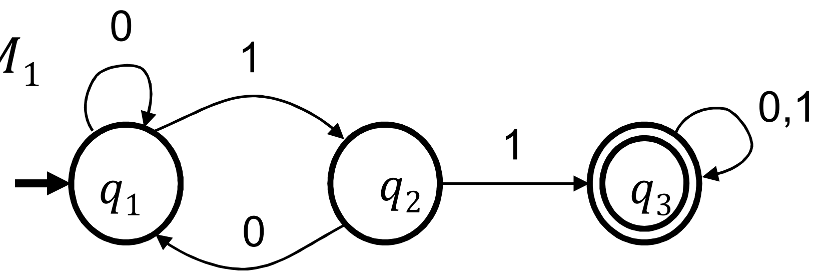
		0	1
$\rightarrow$	q_1	q_1	q_2
	q_2	q_1	q_3
$*$	q_3	q_3	q_3

Arrow for start state

Final states starred

Rows = states

M_1



Strings Accepted by a DFA

- An DFA accepts a string $w = a_1 a_2 \ldots a_n$ if its path in the transition diagram that
  1. Begins at the start state
  2. Ends at an accepting state

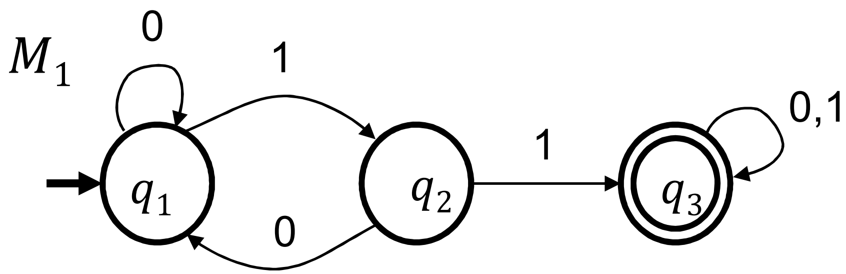
- This DFA accepts input: 01101

$$q_1 \xrightarrow{0} q_1 \xrightarrow{1} q_2 \xrightarrow{1} q_3 \xrightarrow{0} q_3 \xrightarrow{1} q_3$$

- This DFA does not accept input: 00101

$$q_1 \xrightarrow{0} q_1 \xrightarrow{0} q_1 \xrightarrow{1} q_2 \xrightarrow{0} q_1 \xrightarrow{1} q_2$$

- What about 0000?



Language Accepted by a DFA

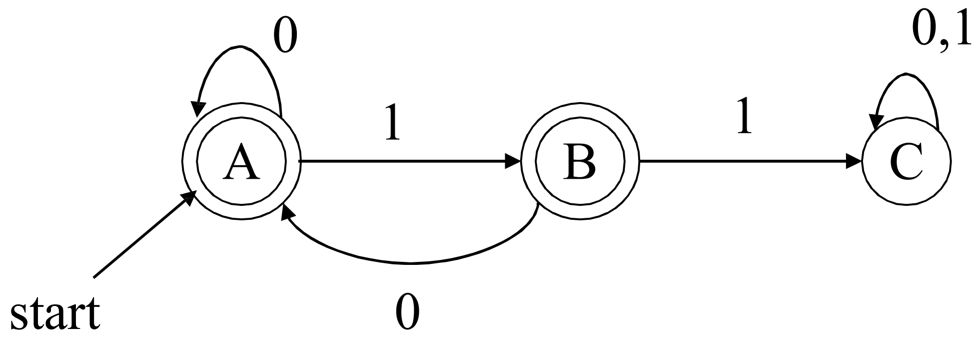
- Informally, the language A, accepted by a DFA M_1, is the set of all strings that are recognized by M_1 ($A = L(M_1)$).

- **Formally, the language accepted by a DFA is $L(M_1)$ such that**

$$L(M_1) = \{ w \mid \delta(q_0, w) \in F \}$$

where $\mathbf{q_0}$ is the starting state of M_1 and F is the final states of M_1

Language Accepted by a DFA

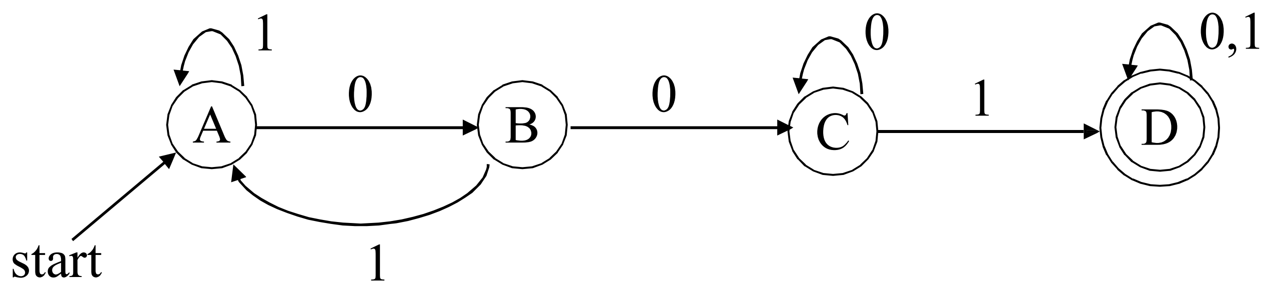


- This DFA accepts all strings of 0's and 1's without two consecutive 1's.
- Formally,

  L(A) = { w | w is in {0,1}* and w does not have two consecutive 1's }

DFA Examples

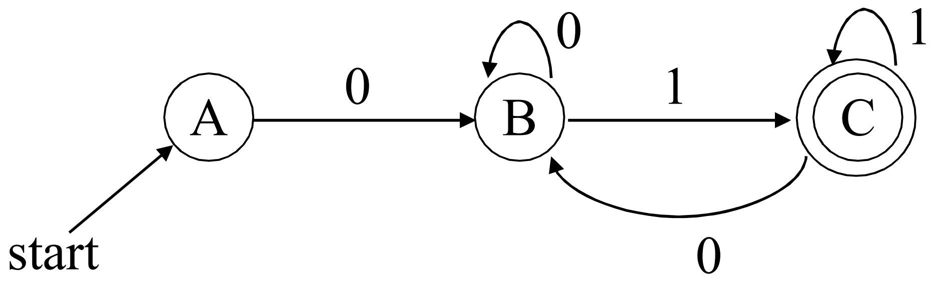
- **A DFA accepting all strings of 0's and 1's containing 001.**



- What do states represent?
  - A: (empty string) OR (strings do not contain 001 and end in 1)
  - B: (string 0) OR (strings do not contain 001 and end in 10)
  - C: strings do not contain 001 and end in 00
  - D: strings contain 001

DFA Examples

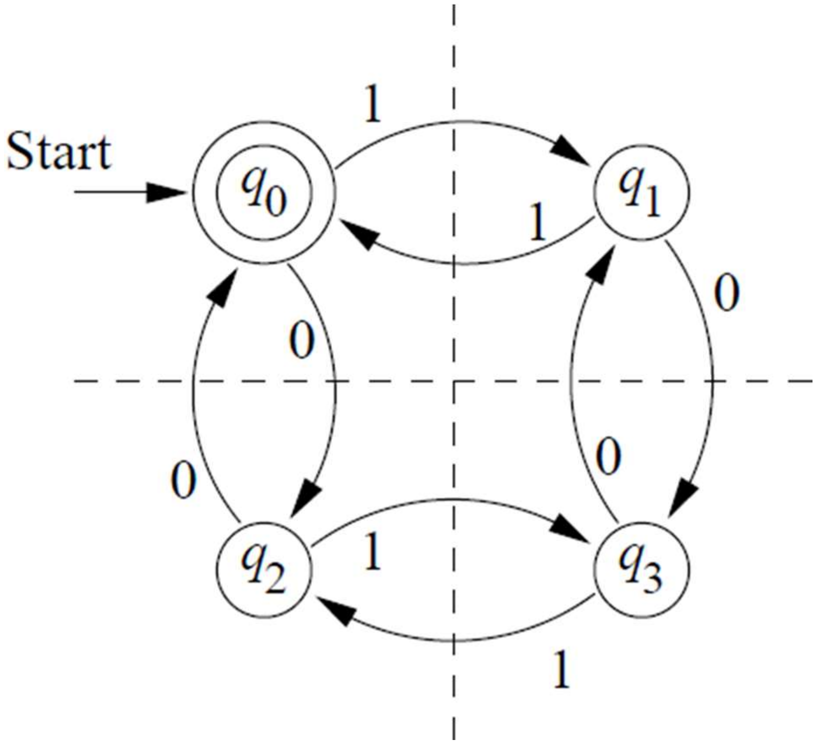
- **A DFA accepting all strings of 0's and 1's which start with 0 and end in 1.**



- What do states represent?
  - A: empty string
  - B: strings start with 0 and end in 0
  - C: strings start with 0 and end in 1

DFA Examples

- **A DFA accepting all and only strings with an even number of 0's and an even number of 1's**



What do states represent?

- q_0: strings with an even number of 0's and an even number of 1's
- q_1: strings with an even number of 0's and an odd number of 1's
- q_2: strings with an odd number of 0's and an even number of 1's
- q_3: strings with an odd number of 0's and an odd number of 1's

DFA Exercises

- Give DFA’s accepting the following languages over the alphabet {0,1}.

1. The set of all strings ending in 00.
2. The set of all strings. i.e. $\{0,1\}^*$
3. The set of all non-empty strings. i.e. $\{0,1\}^+$
4. The empty language. i.e. {}
5. The language that contains only the empty string. i.e. the set $\{\varepsilon\}$
6. The language $\{ 0^n1^k \mid n \geq 1 \text{ and } k \geq 1\}$
7. The strings whose second characters from the right end are 1.
8. The strings whose third characters from the right end are 1.

Regular Languages

- A language L is **regular** if it is the language **accepted by some DFA**.
  - A language is **regular** if it can be described by a **regular expression**.
- Some languages are **not regular**.
  - If a language is **not regular**, there is **no DFA for that language.**

Example:

- $L_1 = \{0^n 1^n \mid n \geq 1\}$ is not regular.
- The set of strings consisting of n 0's followed by n 1's, such that n is at least 1.
- Thus, $L_1 = \{01, 0011, 000111, \ldots\}$

DFA and Regular Languages

- **Every DFA recognizes a regular language, and there is a DFA for every regular language.**

DFA ⟷ Regular Languages

- Some languages are **not regular**. If a language is **not regular**, there is **no DFA for that language.**

Takeaway:

- **Languages accepted by DFAs** are called as **regular languages**.
  - Every DFA accepts **a regular language**, and
  - For every **regular language** there is a DFA that accepts it

Regular Operations

- Let A and B be languages.

  Union: $A \cup B = \{w \mid w \in A \text{ or } w \in B\}$

  Concatenation: $A \circ B = \{xy \mid x \in A \text{ and } y \in B\} = AB$

  Star: $A^* = \{x_1 x_2 \ldots x_k \mid \text{each } x_i \in A \text{ for } k \geq 0\}$

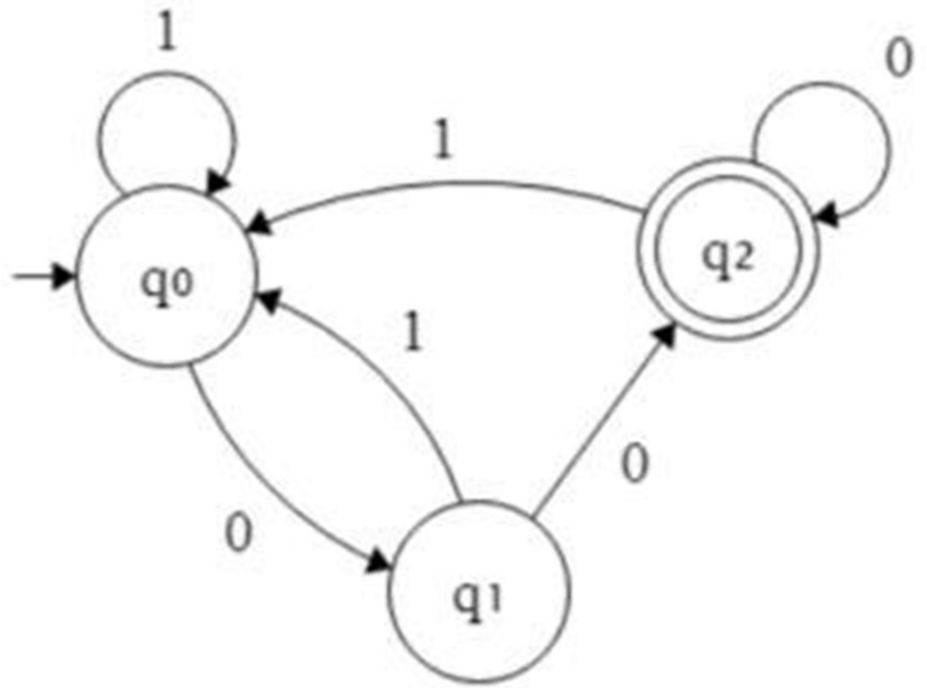
Example:

Let $A = \{\text{good, bad}\}$ and $B = \{\text{boy, girl}\}$.

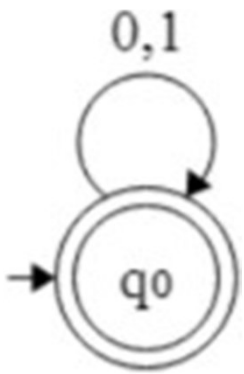
- $A \cup B = \{\text{good, bad, boy, girl}\}$
- $A \circ B = AB = \{\text{goodboy, goodgirl, badboy, badgirl}\}$
- $A^* = \{\varepsilon, \text{good, bad, goodgood, goodbad, badgood, badbad, goodgoodgood, goodgoodbad, } \ldots\}$

- The class of regular languages is **closed** under the **union, concatenation and star operation.**

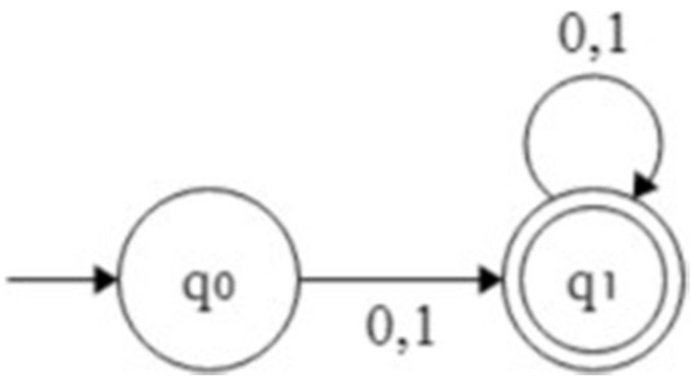
Language over alphabet {0,1}: The set of all strings ending in 00



Language over alphabet {0,1}: The set of all strings. i.e. {0,1}*

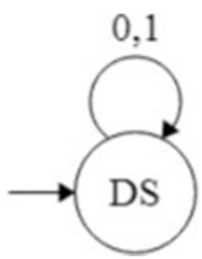


Language over alphabet {0,1}: The set of all non-empty strings. i.e. $\{0,1\}^+$

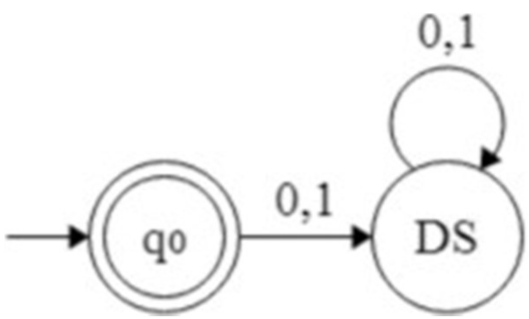


Languages over alphabet {0,1}

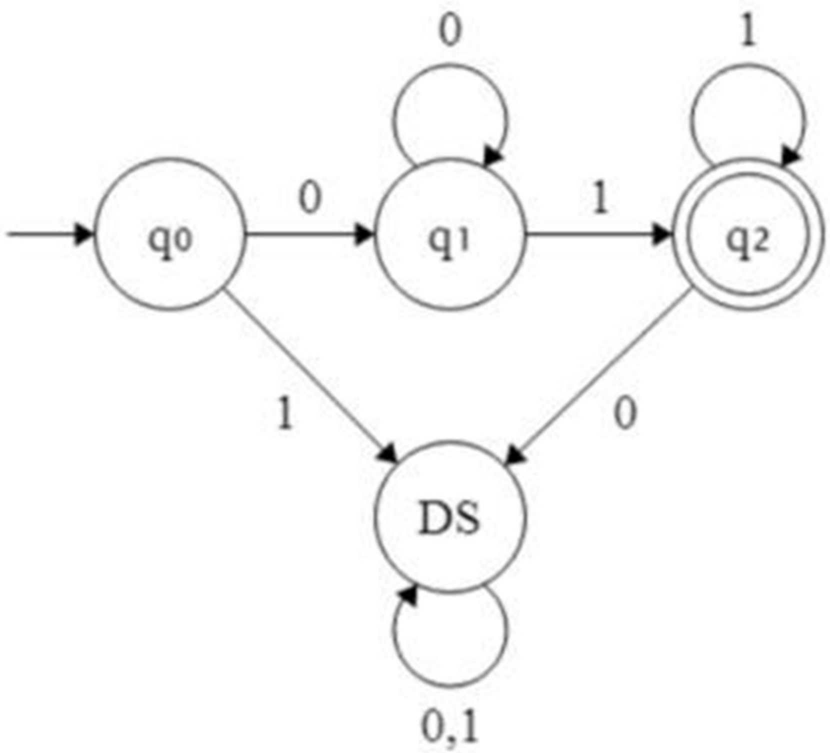
The empty language. i.e. {}



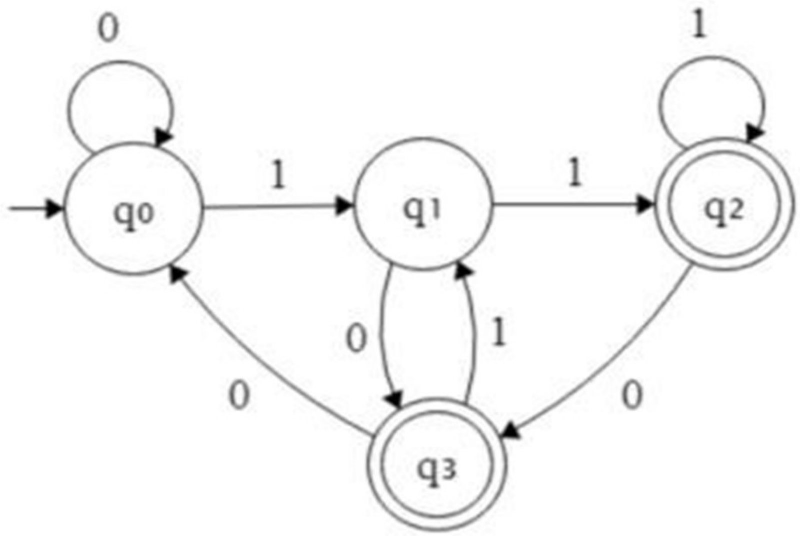
The language that contains only the empty string. i.e. the set {ε}



Language over alphabet {0,1}: The language $\{ 0^n 1^k \mid n \geq 1 \text{ and } k \geq 1 \}$



Languages over the alphabet {0,1}: The strings whose second characters from the right end are 1.



Languages over the alphabet {0,1}: The strings whose third characters from the right end are 1.