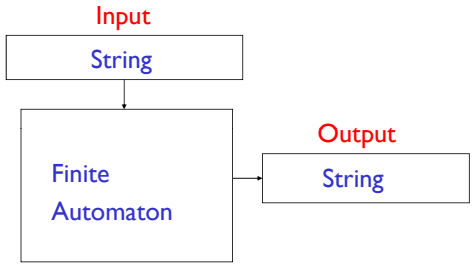
Department of Computer & Information Sciences
Pakistan Institute of Engineering and Applied Sciences

# Finite Automata

Based on slides by Costas Busch [http://csc.lsu.edu/~busch] Umar Faiz

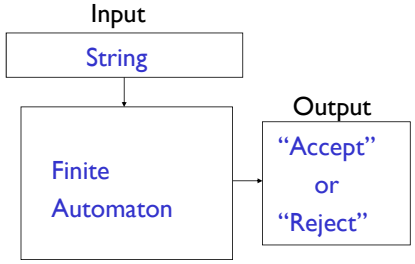
http://www.pieas.edu.pk/umarfaiz/cis317

## Finite Automaton





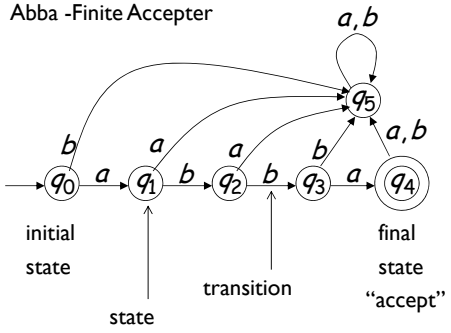

## Finite Accepter





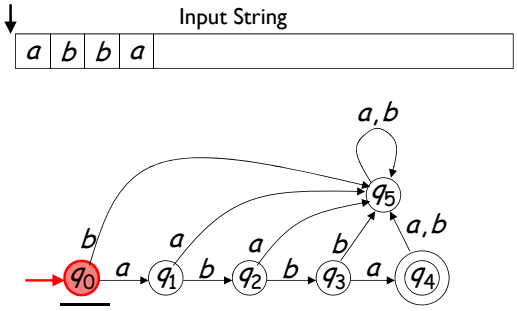

## Transition Graph

Abba -Finite Accepter



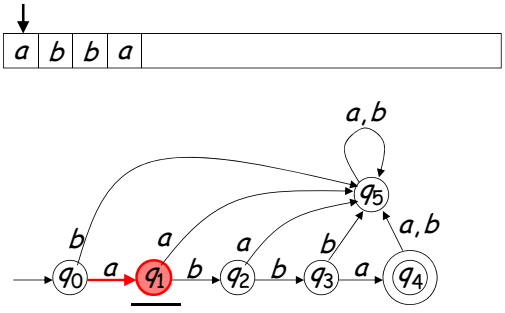



## Initial Configuration

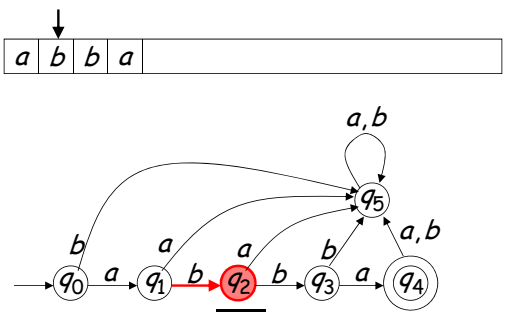





## Reading the Input

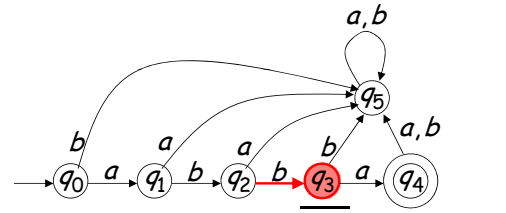


## Reading the Input



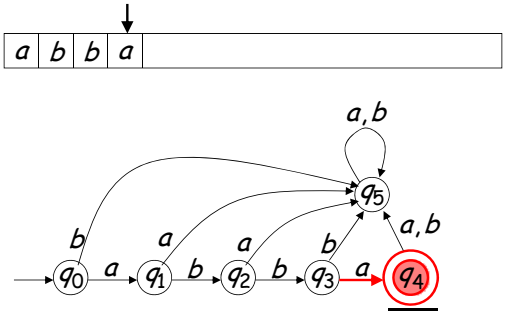



## Reading the Input



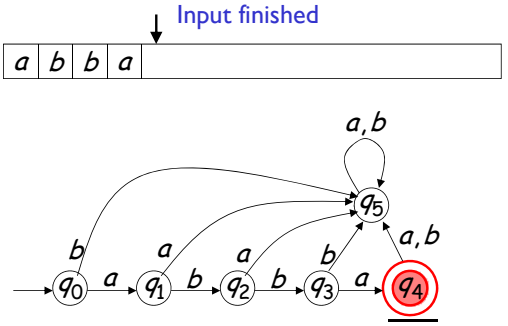



## Reading the Input





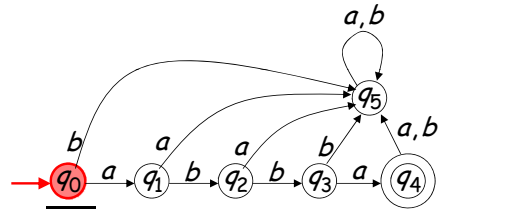

## Input finished



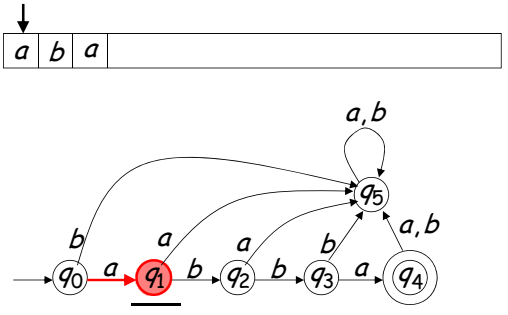

Output: "accept"

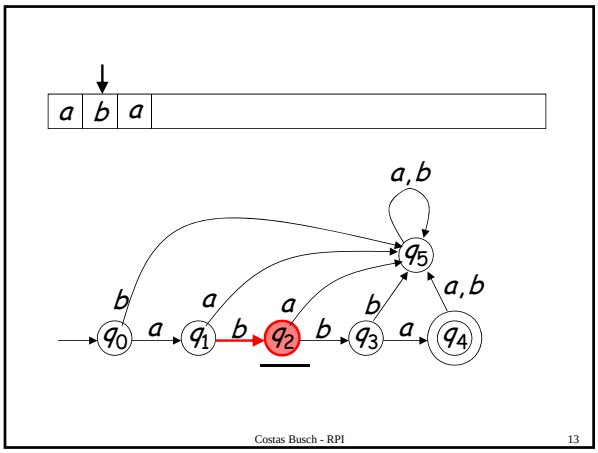


## Rejection

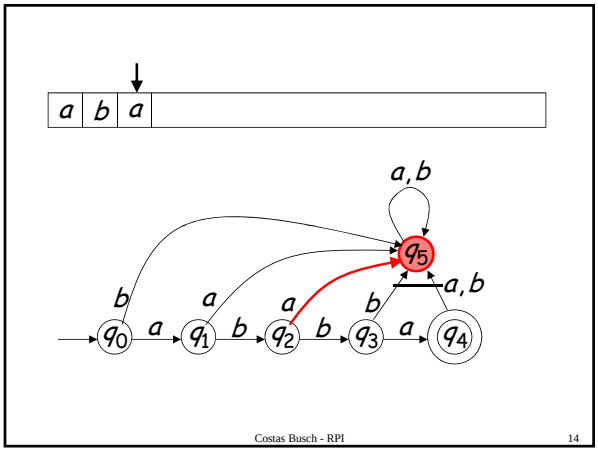




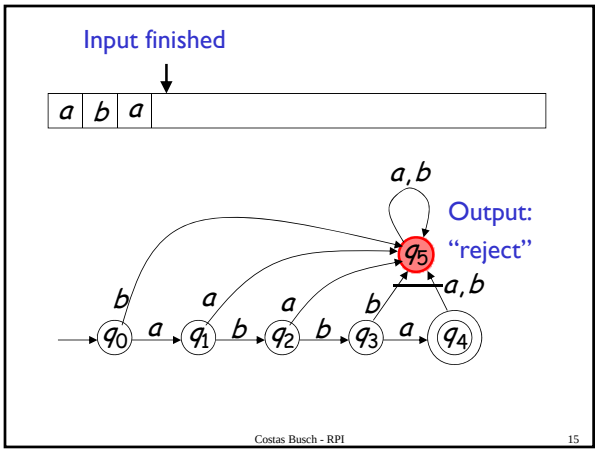




a b a
a,b
b
a
a
b
a,b
$q_0$ $q_1$ $q_2$ $q_3$ $q_4$ $q_5$

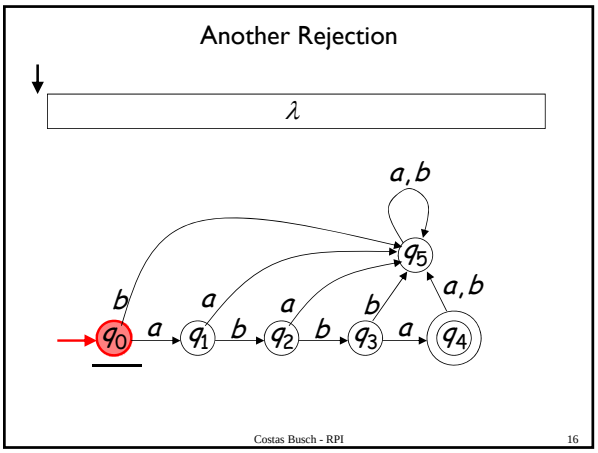



a b a
a,b
b
a
a
b
a,b
$q_0$ $q_1$ $q_2$ $q_3$ $q_4$ $q_5$

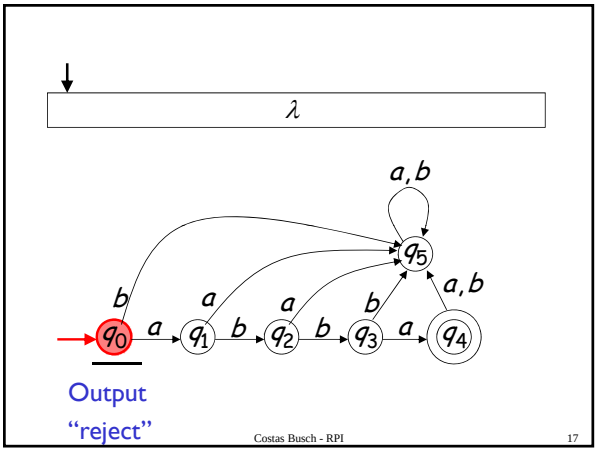



Input finished
a b a
a,b
Output:
"reject"
b
a
a
b
a,b
$q_0$ $q_1$ $q_2$ $q_3$ $q_4$ $q_5$

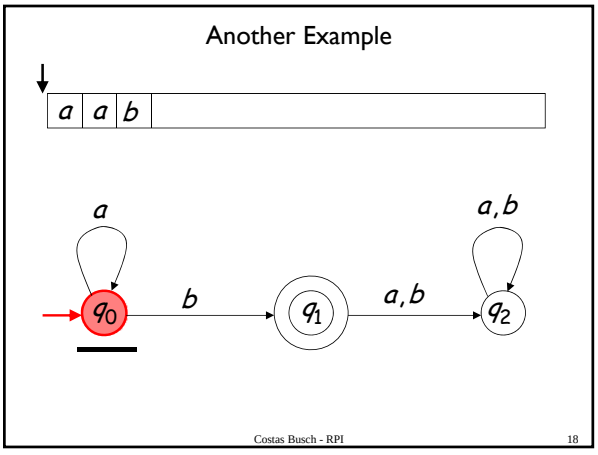


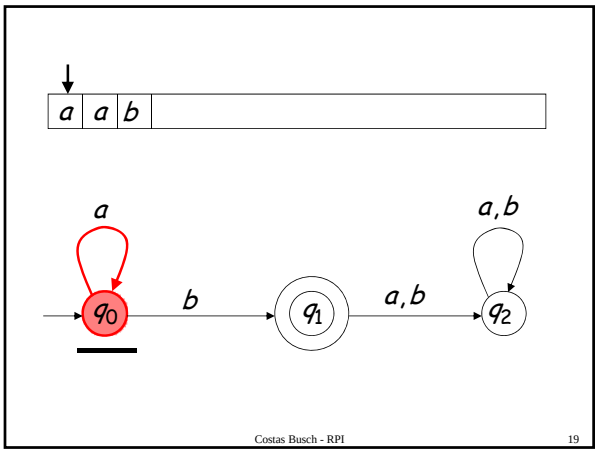

Another Rejection
λ
a,b
b
a
a
b
a,b
$q_0$ $q_1$ $q_2$ $q_3$ $q_4$ $q_5$

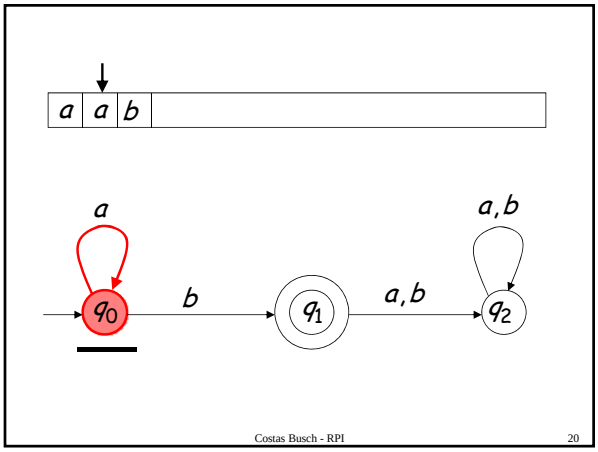


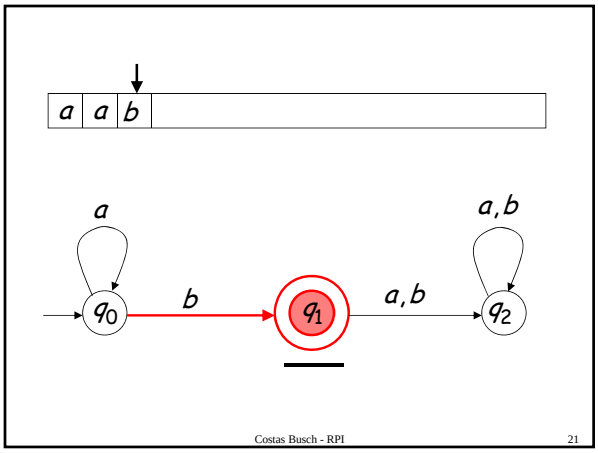

λ
a,b
b
a
a
b
a,b
$q_0$ $q_1$ $q_2$ $q_3$ $q_4$ $q_5$
Output
"reject"

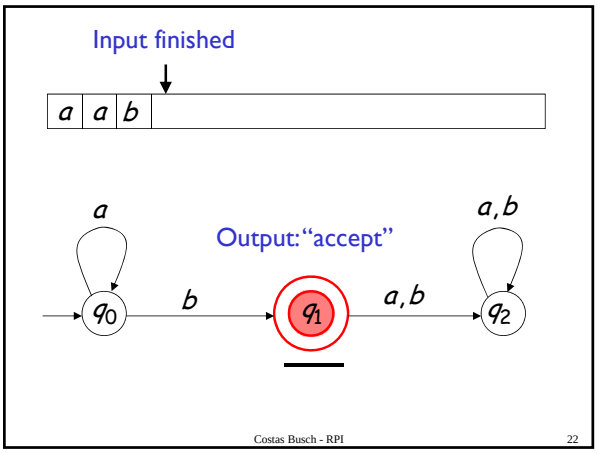


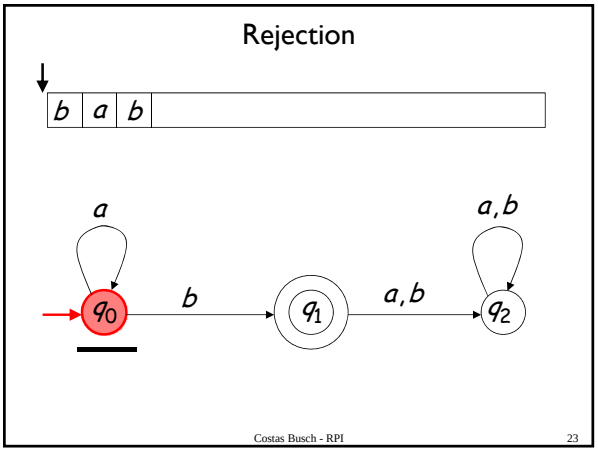

Another Example
a a b
a
a,b
b
$q_0$ $q_1$ $q_2$

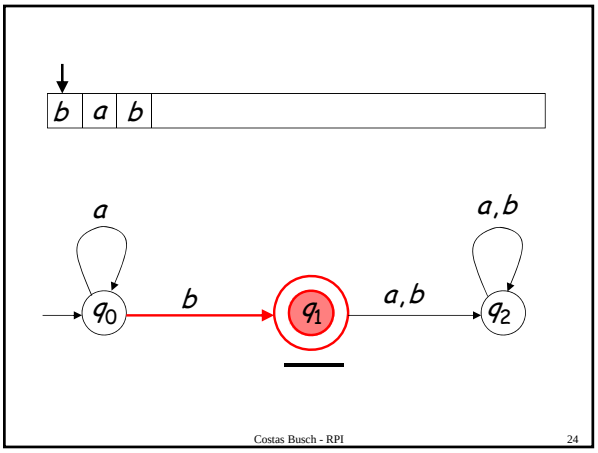

a a b
a
a, b
b
a, b
$q_0$
$q_1$
$q_2$

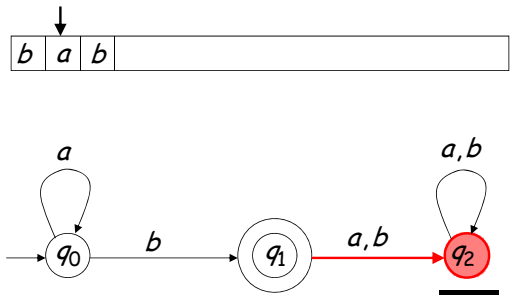


a a b
a
a, b
b
a, b
$q_0$
$q_1$
$q_2$



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

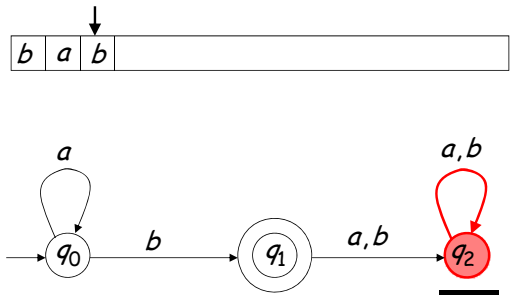


Input finished
a a b
a
Output: "accept"
a, b
b
a, b
$q_0$
$q_1$
$q_2$



Rejection
b a b
a
a, b
b
a, b
$q_0$
$q_1$
$q_2$



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

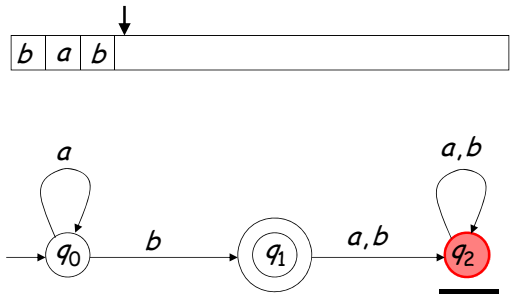







Input finished




Output: "reject"



## Introduction

- Two types of finite automata (FA):
  - Deterministic FA (DFA)
  - Nondeterministic FA (NFA)
- Both types are of the same power,
  - but the latter is more efficient for descriptions of applications of FA,
  - while the former is easier for hardware and software implementations.



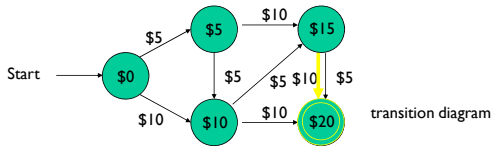
## An Informal Picture of Finite Automata

- A complete application example of finite automata for protocol design and verification
- Read by yourself!
- Involving a concept of "product of two automata"



## Deterministic Finite Automata

- Recall the example of a vending machine selling 20-dolllar food in Chapter 0



## Deterministic Finite Automata

Definition of DFA

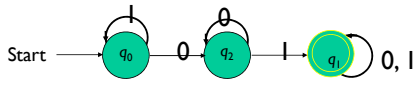
- A DFA A consists of
  - a finite (nonempty) set of states Q;
  - a finite (nonempty) set of input symbols S;
  - a (state) transition function d such that
- d(q, a) = p
  - means
- "automaton A, in state q, takes input a and enters
- state p;"
  - a start state q0;
  - a set of (nonempty) final or accepting states F.
- A may be written as a "5-tuple" A = (Q, S, d, q0, F).

31

## Deterministic Finite Automata

Simpler Notations for DFA's

- Transition diagram




- Transition table

| | 0 | 1 |
|---|---|---|
| $\rightarrow q_0$ | $q_2$ | $q_0$ |
| $^*q_1$ | $q_1$ | $q_1$ |
| $q_2$ | $q_2$ | $q_1$ |

34

## Deterministic Finite Automata

How a DFA Processes Strings

- Given an input string x = a1a2…an, if
- d(q0, a1) = q1, d(q1, a2) = q2, …,
- d(qi–1, ai) = qi, ..., d(qn–1, an) = qn,
- and qn ∈ F,
- then x is accepted; otherwise, "rejected."
- Every transition is deterministic.

32

## Deterministic Finite Automata

Extending Transition Function to Strings

- Extended transition function
- If x = a1a2…an and d is such that
- d(p, a1) = q1, d(q1, a2) = q2, …,
- d(qi–1, ai) = qi, ..., d(qn–1, an) = q,
- then we define to be
- (p, x) = q.
- Also may be defined recursively as in the textbook.

35

## Deterministic Finite Automata

Example: Design an FA A to accept the language

- L = {x01y | x and y are any strings of 0's and 1's}.

Examples of strings in L:

- 01, 11010, 100011…

Transitions:

- d(q0, 1) = q0, d(q0, 0) = q2, d(q2, 1) = q1, d(q2, 0) = q2,
- d(q1, 0) = q1, d(q1, 1) = q1
- 5-Tuple:
- A = ({q0, q1, q2}, {0, 1}, d, q0, {q1})

33

## Deterministic Finite Automata

Extending Transition Function to Strings

- Recursive definition for
- Basis: (q, e) = q.
- Induction: if w = xa (a is last symbol of w), then
- (q, w) = d( (q, x), a).

36

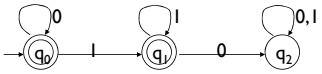
## Deterministic Finite Automata

The Language of a DFA

- The language of a DFA A is defined as
- L(A) = {w | (q0, w) is in F}.
- If L is L(A) for some DFA A, then we say L is a regular language.



## Example




alphabet S = {0, 1}
start state Q = {$q_0$, $q_1$, $q_2$}
initial state $q_0$
accepting states F = {$q_0$, $q_1$}

transition function δ:

| states \ inputs | 0 | 1 |
|---|---|---|
| $q_0$ | $q_0$ | $q_1$ |
| $q_1$ | $q_2$ | $q_1$ |
| $q_2$ | $q_2$ | $q_2$ |

## Formalities

Deterministic Finite Accepter (DFA)

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

$Q$ : set of states

$\Sigma$ : input alphabet

$\delta$ : transition function

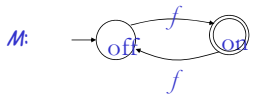
$q_0$ : initial state

$F$ : set of final states



## Language of a DFA

The language of a DFA $(Q, \Sigma, \delta, q_0, F)$ is the set of all strings over $\Sigma$ that, starting from $q_0$ and following the transitions as the string is read left to right, will reach some accepting state.

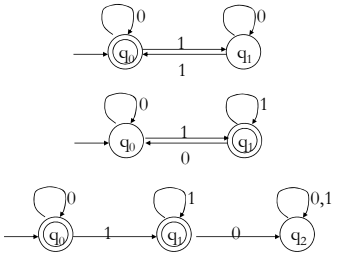



- Language of $M$ is $\{f, fff, fffff, \ldots\} = \{f^n : n \text{ is odd}\}$

## Deterministic Finite Automata

- A deterministic finite automaton (DFA) is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where
  - $Q$ is a finite set of states
  - $\Sigma$ is an alphabet
  - $\delta: Q \times \Sigma \rightarrow Q$ is a transition function
  - $q_0 \in Q$ is the initial state
  - $F \subseteq Q$ is a set of accepting states (or final states).
- In diagrams, the accepting states will be denoted by double loops

## Examples




What are the languages of these DFAs?

## Examples

- Construct a DFA that accepts the language

L = {010, 1}  ( S = {0, 1} )

## Examples

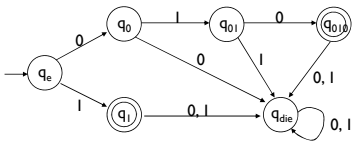
- Construct a DFA over alphabet $\{0, 1\}$ that accepts all strings that end in $101$
- Hint: The DFA must "remember" the last 3 bits of the string it is reading

## Examples

- Construct a DFA that accepts the language

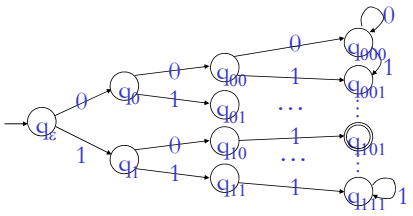
L = {010, 1}  ( S = {0, 1} )

- Answer




## Examples

- Construct a DFA over alphabet $\{0, 1\}$ that accepts all strings that end in $101$
- Sketch of answer:

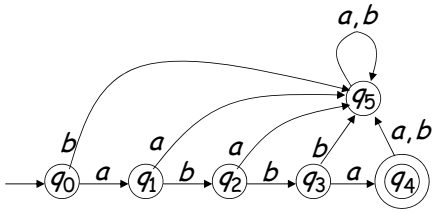



## Examples

- Construct a DFA over alphabet $\{0, 1\}$ that accepts all strings that end in $101$

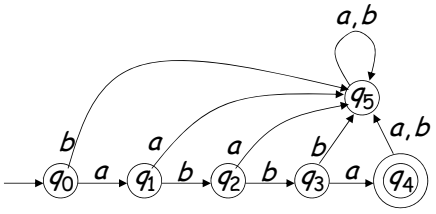
## Input Alphabet $\Sigma$

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



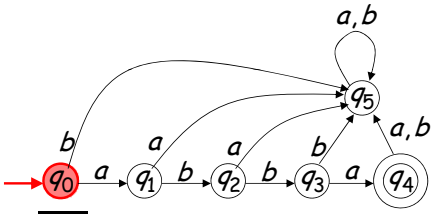
Set of States $Q$

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





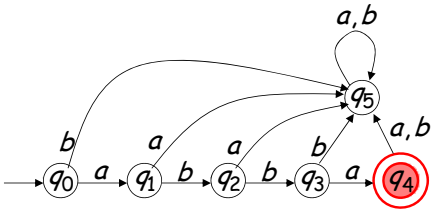

Initial State $q_0$






Set of Final States $F$

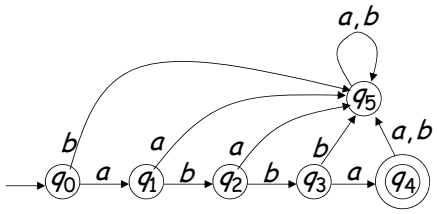
$F = \{q_4\}$





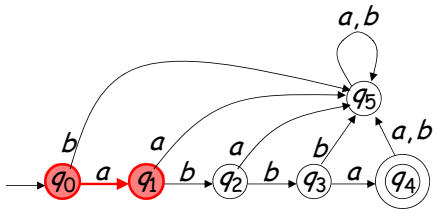

Transition Function $\delta$

$\delta : Q \times \Sigma \to Q$



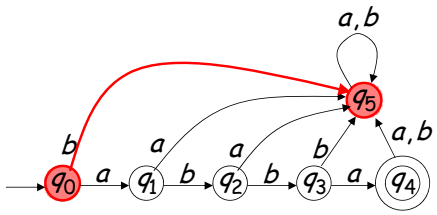



$\delta(q_0, a) = q_1$

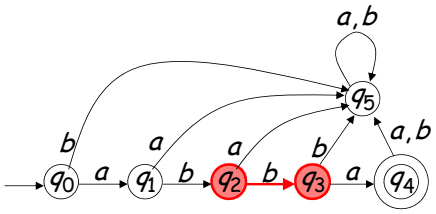





$\delta(q_0, b) = q_5$



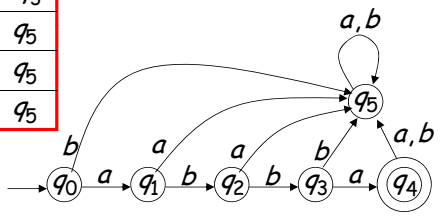
$\delta(q_2, b) = q_3$






Transition Function $\delta$

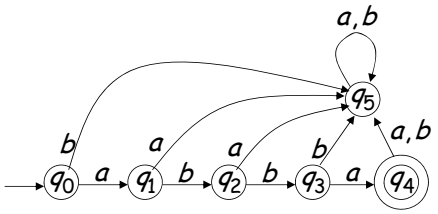
| $\delta$ | $a$ | $b$ |
|---|---|---|
| $q_0$ | $q_1$ | $q_5$ |
| $q_1$ | $q_5$ | $q_2$ |
| $q_2$ | $q_5$ | $q_3$ |
| $q_3$ | $q_4$ | $q_5$ |
| $q_4$ | $q_5$ | $q_5$ |
| $q_5$ | $q_5$ | $q_5$ |





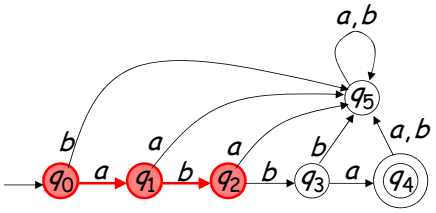

Extended Transition Function $\delta^*$

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



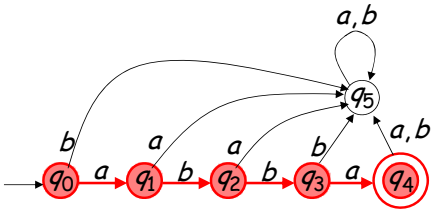



$\delta^*(q_0, ab) = q_2$



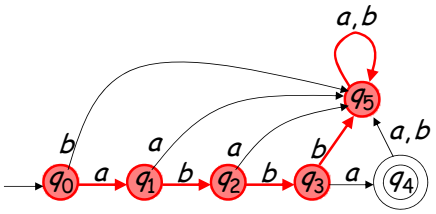



$\delta^*(q_0, abba) = q_4$



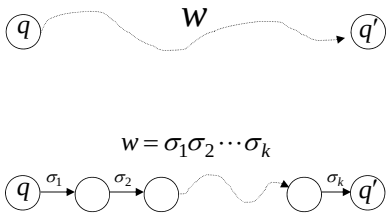



$\delta^*(q_0, abbbaa) = q_5$



Observation: There is a walk from $q$ to $q'$ with label $w$

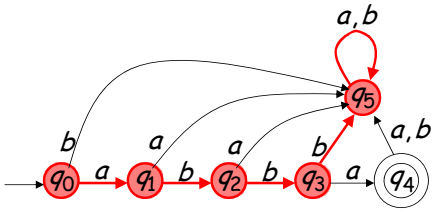
$$\delta^*(q,w)=q'$$






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

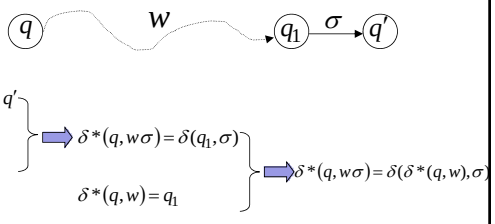
$$\delta^*(q_0,abbbaa)=q_5$$





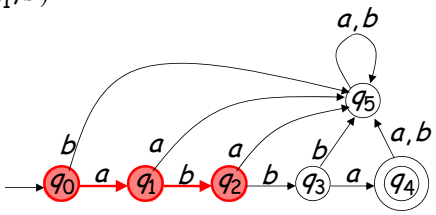

Recursive Definition

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






$\delta^*(q_0,ab)=$
$\delta(\delta^*(q_0,a),b)=$
$\delta(\delta(\delta^*(q_0,\lambda),a),b)=$
$\delta(\delta(q_0,a),b)=$
$\delta(q_1,b)=$
$q_2$






Languages Accepted by DFAs

Take DFA $M$

Definition:

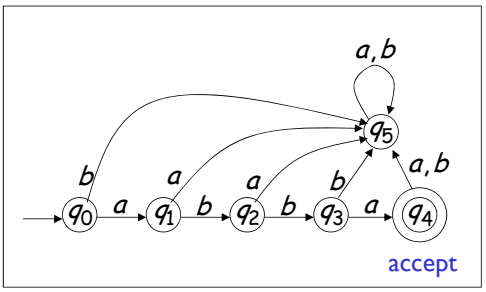
The language $L(M)$ contains all input strings accepted by $M$

$L(M)$ = { strings that drive $M$ to a final state}



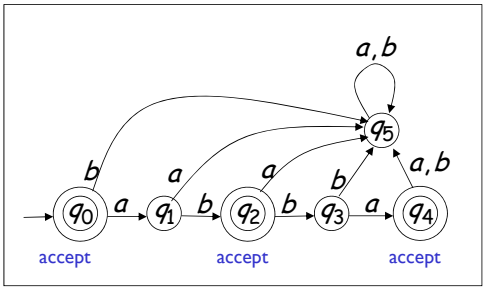
Example

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



## Another Example

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






## Formally

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

Language accepted by : $M$

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




## Observation

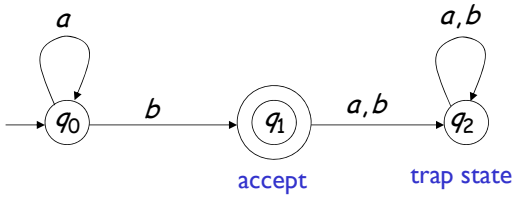
Language rejected by : $M$

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



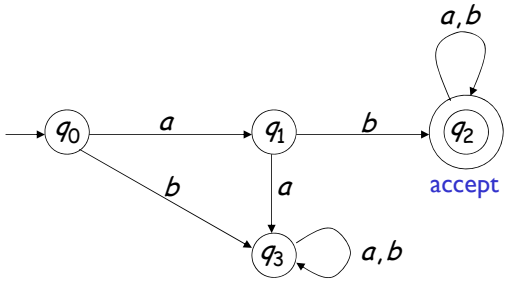

## More Examples

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



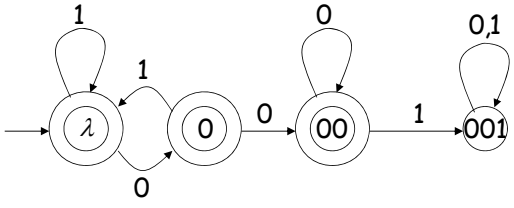



$L(M)$= { all strings with prefix }$ab$






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



## Regular Languages

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

All regular languages form a language family



## Examples of regular languages:

$\{abba\}$ $\{\lambda, ab, abba\}$ $\{a^n b : n \geq 0\}$

{ all strings with prefix } $ab$

{ all strings with prefix } $ab$

{ all strings without substring } $001$

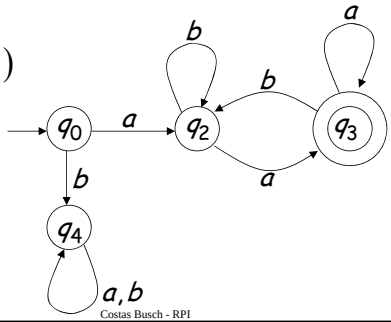
There exist automata that accept these Languages (see previous slides).



## Another Example

The language is regular: $L = \{awa : w \in \{a,b\}^*\}$

$L = L(M)$






There exist languages which are not Regular:

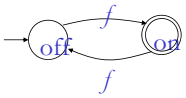
Example: $L = \{a^n b^n : n \geq 0\}$

There is no DFA that accepts such a language

(we will prove this later in the class)



## 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: $\{f, fff, fffff, \ldots\} = \{f^n : n \text{ is odd}\}$

This is an example of a deterministic finite automaton over alphabet $\{f\}$

## Deterministic finite automata

A deterministic finite automaton (DFA) is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where

$Q$ is a finite set of states

$\Sigma$ is an alphabet

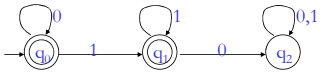
$\delta$: $Q \times \Sigma \rightarrow Q$ is a transition function

$q_0 \in Q$ is the initial state

$F \subseteq Q$ is a set of accepting states (or final states).

In diagrams, the accepting states will be denoted by double loops

## Example




alphabet $\Sigma = \{0, 1\}$
start state $Q = \{q_0, q_1, q_2\}$
initial state $q_0$
accepting states $F = \{q_0, q_1\}$

transition function $\delta$:

| states \ inputs | 0 | 1 |
|---|---|---|
| $q_0$ | $q_0$ | $q_1$ |
| $q_1$ | $q_2$ | $q_1$ |
| $q_2$ | $q_2$ | $q_2$ |

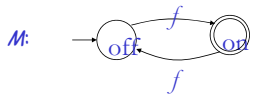
## Examples

Construct a DFA that accepts the language

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

## Language of a DFA

The language of a DFA $(Q, \Sigma, \delta, q_0, F)$ is the set of all strings over $\Sigma$ that, starting from $q_0$ and following the transitions as the string is read left to right, will reach some accepting state.



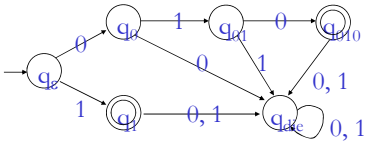

Language of $M$ is $\{f, fff, fffff, \ldots\} = \{f^n: n \text{ is odd}\}$

## Examples

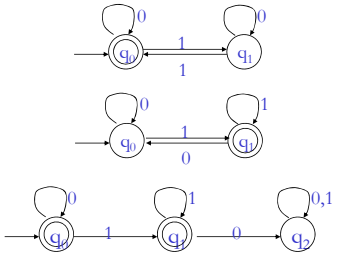
Construct a DFA that accepts the language

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

Answer




## Examples




What are the languages of these DFAs?

## Examples

Construct a DFA over alphabet $\{0, 1\}$ that accepts all strings that end in $101$

## Examples

Construct a DFA over alphabet $\{0, 1\}$ that accepts all strings that end in $101$

Hint: The DFA must "remember" the last 3 bits of the string it is reading

## Examples

Construct a DFA over alphabet $\{0, 1\}$ that accepts all strings that end in $101$

Sketch of answer: