Finite Automata (FA)

Lecture 3.2

Strathmore UNIVERSITY

Strathmore
UNIVERSITY

Lecture Outline

- This lecture covers:
  - Finite automata
  - Finite automata classes
  - Deterministic finite automata
  - Nondeterministic finite automata
  - Finite automata formal definition
  - Applications of finite automata

Finite Automata

Strathmore UNIVERSITY

- Finite Automata, FA (also called Finite State Machine) is a model of computation with **finite set of states** and which acts as a ***language acceptor***
- FA state can change from one to another upon receiving some input symbol
- FA has fixed memory capacity to handle information and does not have **auxiliary memory**

Finite Automata (II)

Strathmore UNIVERSITY

- When FA receives a string input, it can either accept or reject it hence generally regarded as ***language acceptors*** or **language recognition device**
- Any computer whose outputs are either “yes” or “no” acts as a language acceptor
- The language the computer accepts is the set of input strings that cause it to produce the answer yes

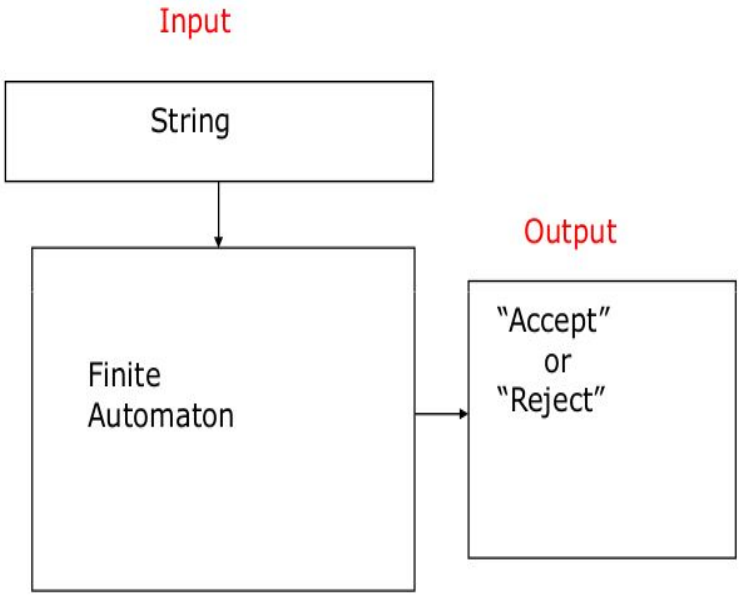
Finite Automata (III)

Strathmore UNIVERSITY

- A language is called a **regular language** if some finite automaton recognizes it
- Due to memory limitation, FA's can only accept **simple languages** which does not require ***remembering more than the current state***

Finite Automata (IV)

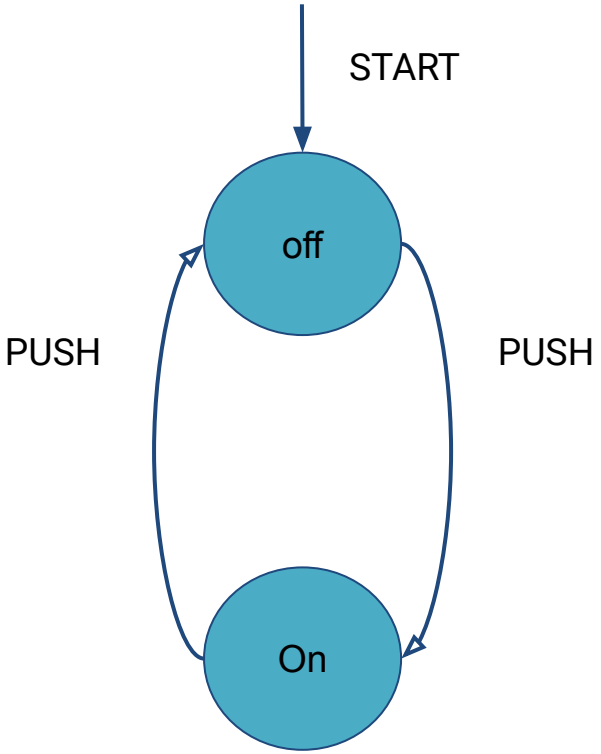
- The machine reads the input string one at a time until the end of string
- If it winds up in one of the set of **final states** the input string is considered to be accepted otherwise **rejected**
- The **language accepted** by the machine is the set of strings it accepts



FA Basic Design

Finite Automata (V)

- FA is the simplest computational model, with very limited memory, but highly useful
- Example: *a finite automaton modeling an on/off switch*



Finite Automata Classifications

Strathmore UNIVERSITY

- Finite automata can be classified into two major categories
  1. Deterministic Finite Automata (DFA)
  2. Nondeterministic Finite Automata (NFA)

Strathmore UNIVERSITY

Deterministic Finite Automata (DFA)

Formal Definition of DFA

Strathmore UNIVERSITY

- DFA is defined by a 5 -tuple:

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

  1. Q: finite set of states
  2. Σ: finite alphabet
  3. δ: transition function, $\delta : Q \times \Sigma \rightarrow Q$, takes a state and input symbol as arguments, and returns a state
  4. $q_0 \in Q$: start state
  5. $F \in Q$: set of accept states

Finite State Diagram

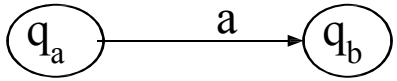
Strathmore UNIVERSITY

- A graphic representation of a finite automaton
- A finite state diagram is a directed graph, where **nodes represent elements** in Q (i.e., states) and **arrows are characters in Σ** such that:

Finite State Diagram (II)

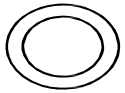
Strathmore UNIVERSITY

- $((q_a,a),q_b)$ is a transition in δ:



- The initial state is marked with:

- The final state(s) are marked with:



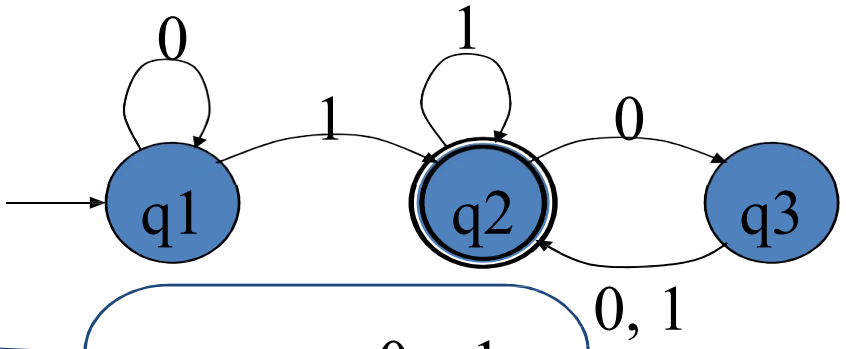
Formal **Description** of DFA - Example

Strathmore UNIVERSITY

FORMAL DESCRIPTION

- M = (Q, Σ, δ, q_0, F), where
  - Q = {q1, q2, q3}
  - Σ = {0, 1}
  - δ(q1,0)=q1, δ(q1,1) = q2, δ (q2,0) = q3, δ(q2,1) = q2, δ (q3,0) = q2, δ(q3,1) = q2
  - q1 is the start state
  - F = {q2}

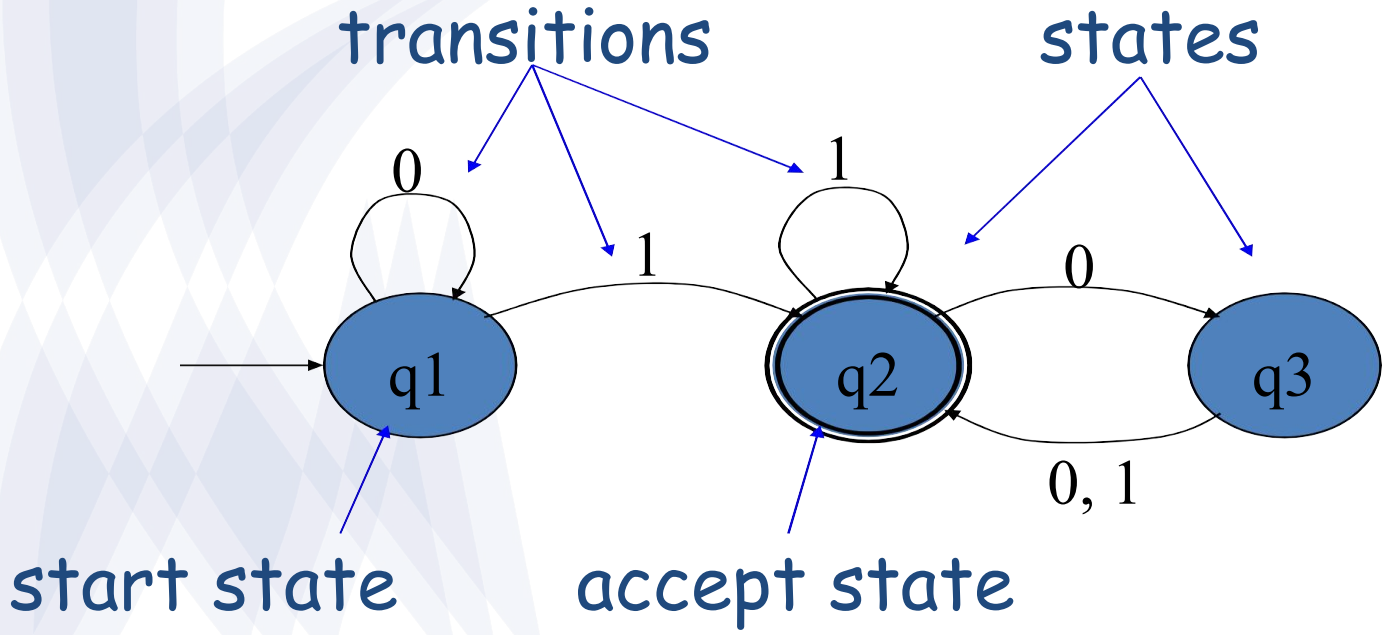
STATE TRANSITION DIAGRAM



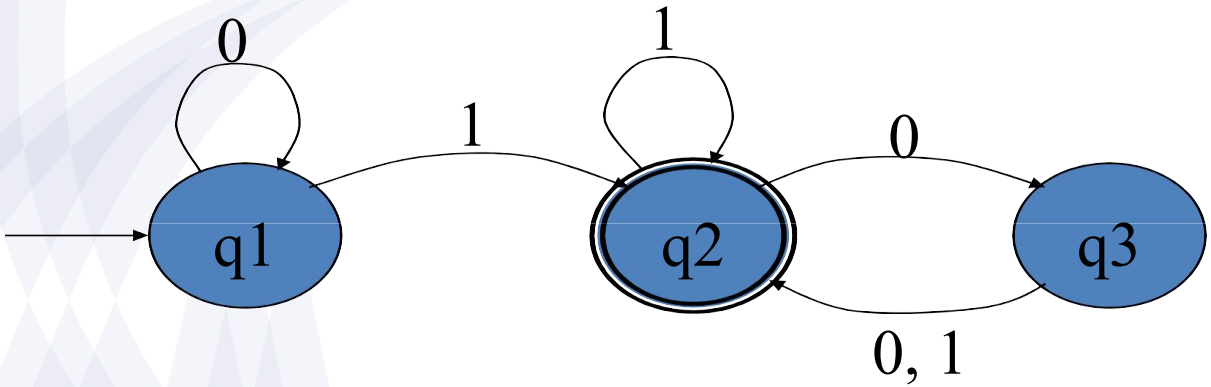
	0	1
q1	q1	q2
q2	q3	q2
q3	q2	q2

STATE TRANSITION TABLE

Finite State Diagram(III)



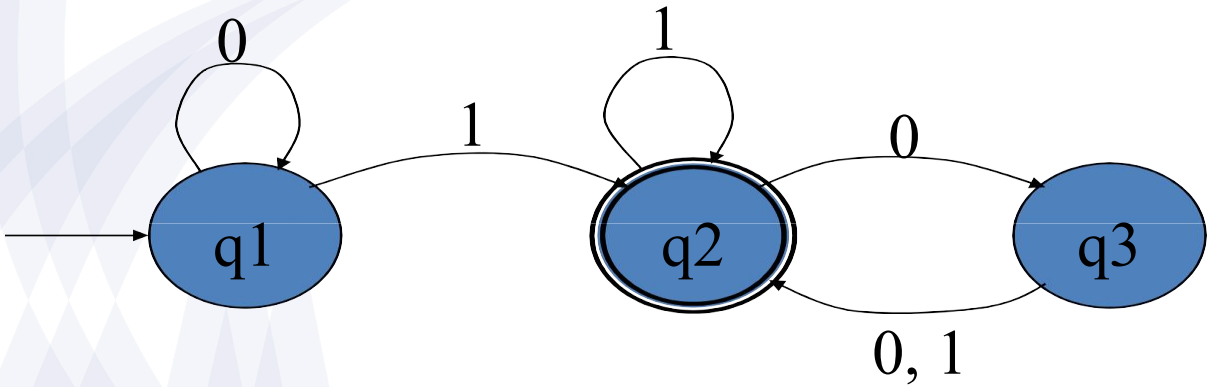
Finite State Diagram(III) - Example 1



- on input “0110”, the machine goes:

  q1→q2→q2→q3 = “reject”

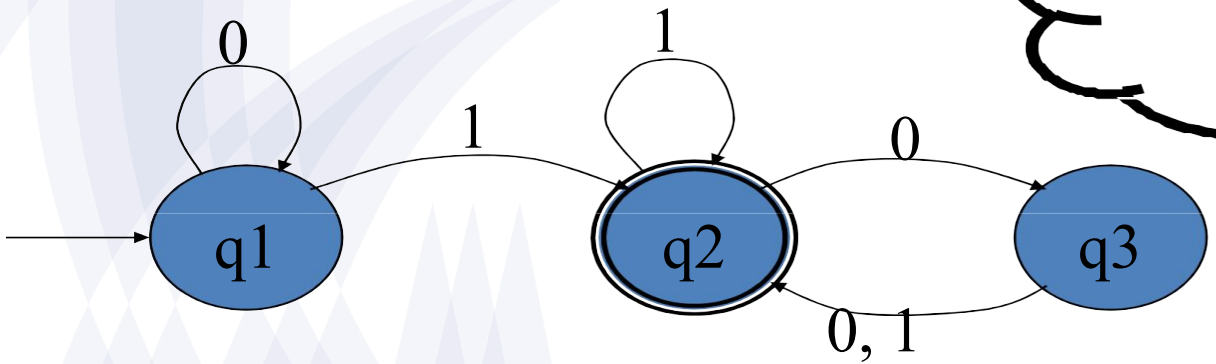
Finite State Diagram(IV) - Example 2



on input “101”, the machine goes:

q1 → q2 → q3 → q2 = “accept”

Finite State Diagram(V) - Example 3



What is the language accepted by this Machine?

010: reject

11: accept

0110100: accept

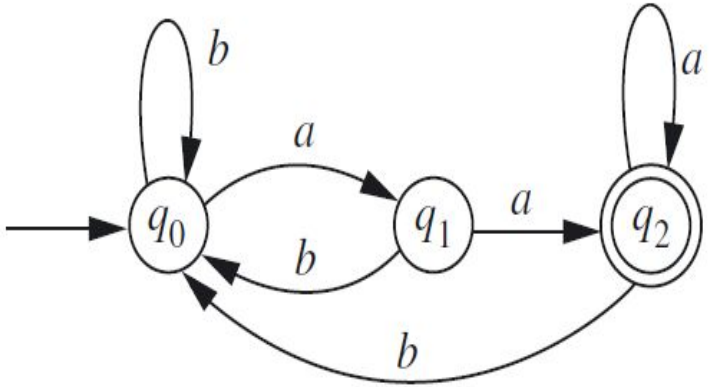
010000010010: reject

Finite State Diagram(V) - Example 4

- Given language L_1 =

$\{x \in \{a, b\}^* \mid x \text{ ends with } aa\}$

- DFA to recognise L_1 (accepting the strings ending with aa)
- What is the transition from start to accept state on
  - aabbaa
  - ababaa
  - aaaaaa

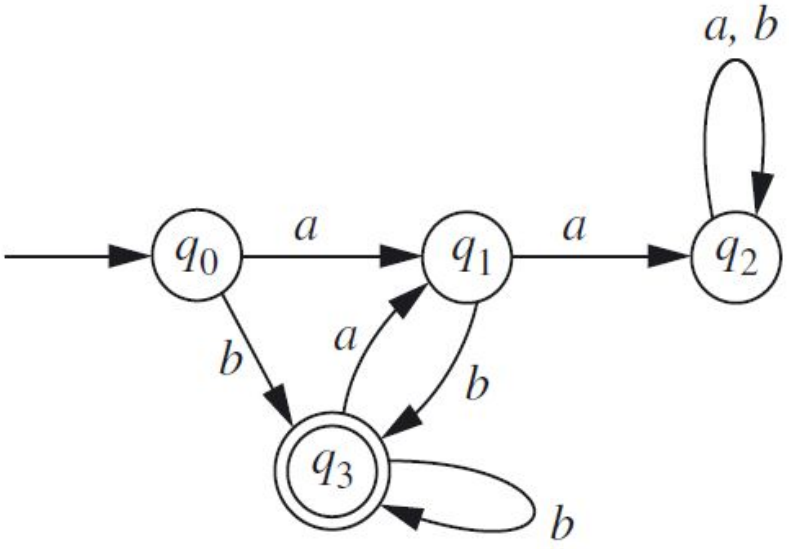


Finite State Diagram(V) - Example 6

- Given language L_2 =

{x ∈ {a, b}* | x ends with b and does not contain the substring aa}

- What is the transition from start to accept state on
  - abab
  - abbbbab



Finite State Machine Language Acceptance

Strathmore UNIVERSITY

- If A is the set of all strings that finite automaton machine **M** accepts, we say that A is the language of machine M and write **L(M) = A.**
- We can also say that *M **recognizes** A or that M **accepts** A*
- A machine may accept several strings, but it always *recognizes only one language.*
- If the machine accepts no strings, it still recognizes one language namely, the empty language ∅

Class Exercise

- SU cafeteria has installed an ice cream vending machine to automatically dispense ice cream to students and staff. The cost of a can of ice cream is Kshs 60 and the machine only accepts coins in the denomination of 20 & 40 only and the machine does not give change.

Exercise (II)

Strathmore UNIVERSITY

a) Formally define this machine as a finite automaton. i.e. in terms of (Q, Σ, δ, q_0, F)

b) Draw the state transition diagram for the machine defined in (a) above

c) Draw the state transition table for the machine defined in (a) and (b) above

Strathmore
UNIVERSITY

Designing Finite Automata

Designing Finite Automata

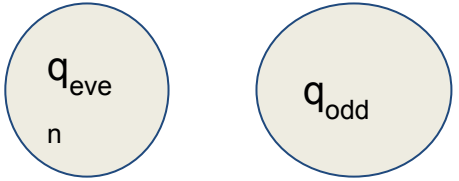
Strathmore UNIVERSITY

- ***Given Σ ={0,1}, L = {w | w has odd number of 1's}, design a FA to recognize L.***
  - What is the necessary information to remember? Is the number of 1's seen so far even or odd?
  - **Two possibilities**

Designing Finite Automata (III)

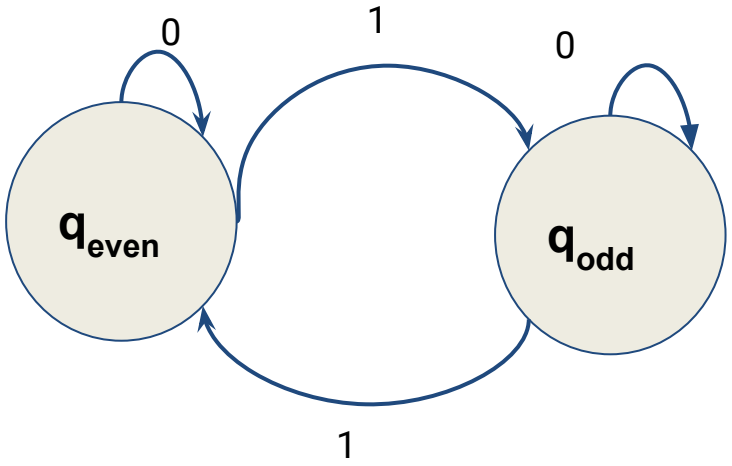
Strathmore UNIVERSITY

- Σ ={0,1}, L = {w | w has odd number of 1‘s}, design a FA to recognize L
- What is the necessary information to remember?
- Is the number of 1’s seen so far even or odd?
- Two possibilities
- **Assign a state to each possibility**



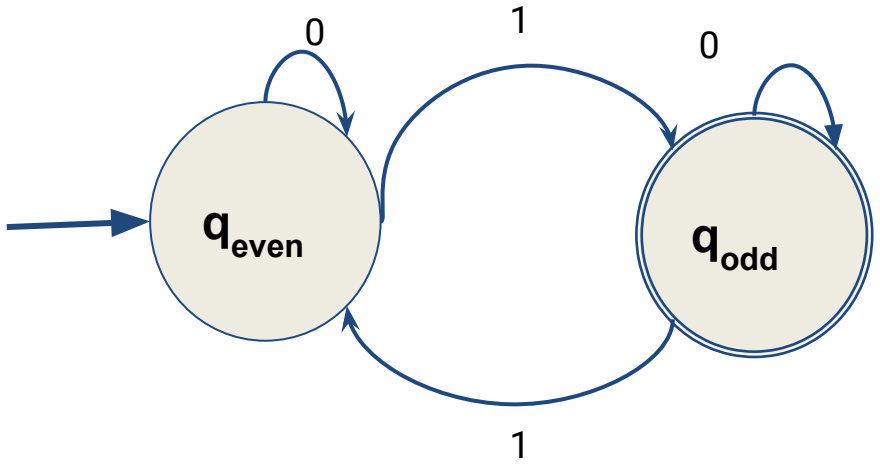
Designing Finite Automata (IV)

- Σ ={0,1}, L = {w | w has odd number of 1‘s}, design a FA to recognize L.
- What is the necessary information to remember?
- Is the number of 1s seen so far even or odd? Two possibilities.
- Assign a state to each possibility.
- **Assign the transitions from one possibility to another upon reading a symbol**



Designing Finite Automata (V)

- Σ ={0,1}, L = {w | w has odd number of 1‘s}, design a FA to recognize L.
- What is the necessary information to remember?
- Is the number of 1s seen so far even or odd? Two possibilities.
- Assign a state to each possibility.
- Assign the transitions from one possibility to another upon reading a symbol
- **Set the start and accept states**

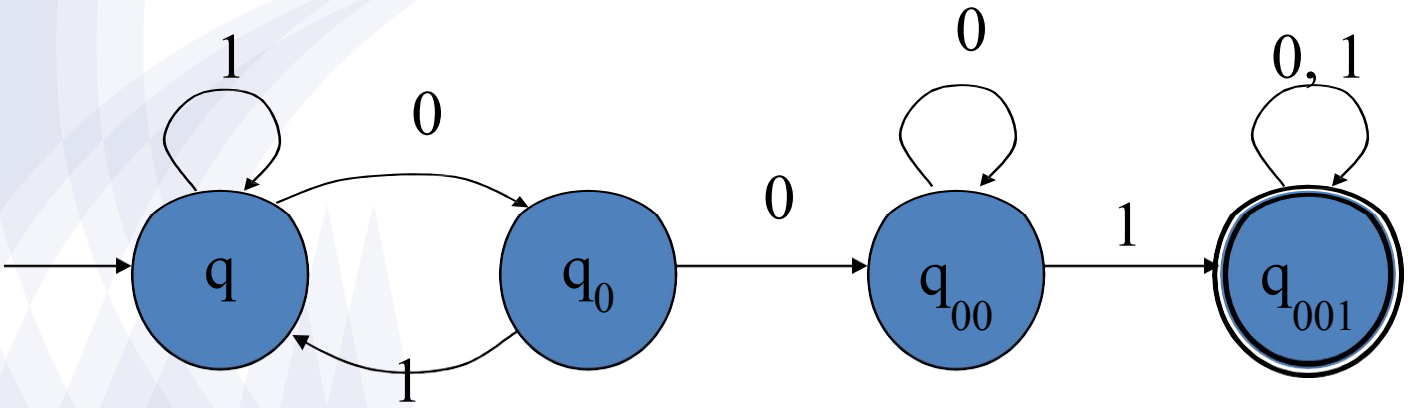


Designing DFA Example

Strathmore UNIVERSITY

- Σ ={0,1}, L = {w | w contains 001 as a substring}, design a DFA to recognize L
- Four possibilities:
    1. haven't seen any symbols of 001 - **q**
    2. have just seen a 0 - $\mathbf{q_0}$
    3. have just seen a 00 - $\mathbf{q_{00}}$
    4. have seen the entire pattern 001 - $\mathbf{q_{001}}$

Designing DFA Example (II)

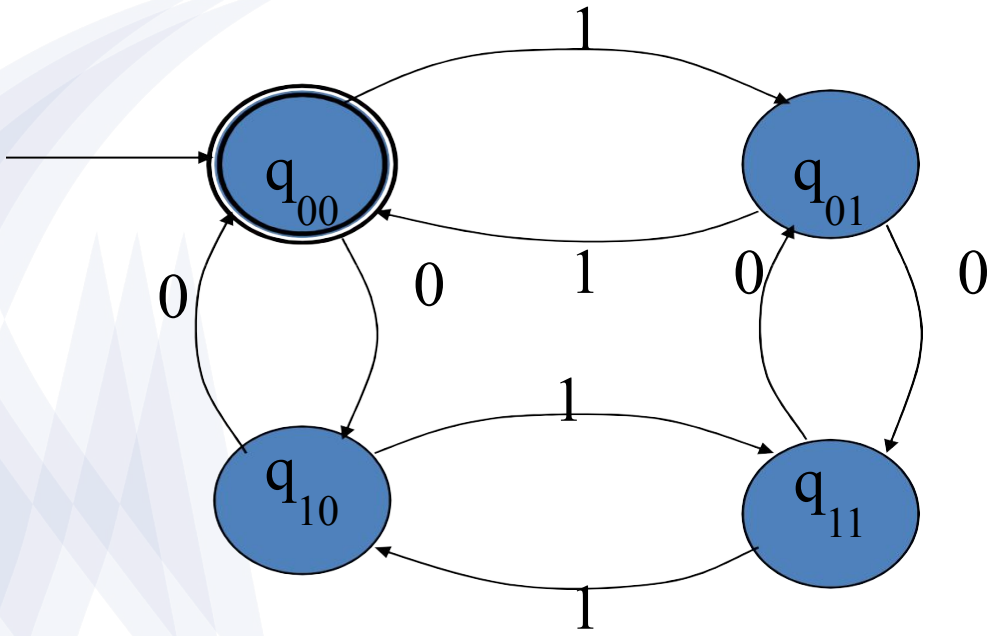


Designing DFA Example (III)

Strathmore UNIVERSITY

- Σ ={0,1}, L = {w | w has even number of 0's and even number of 1's}, design a DFA to recognize L.
- *What to remember?*
- *How many possibilities?*

Designing DFA Example (IV)



Strathmore
UNIVERSITY

Nondeterministic Finite Automata (NFA)

Introduction to NFA

Strathmore UNIVERSITY

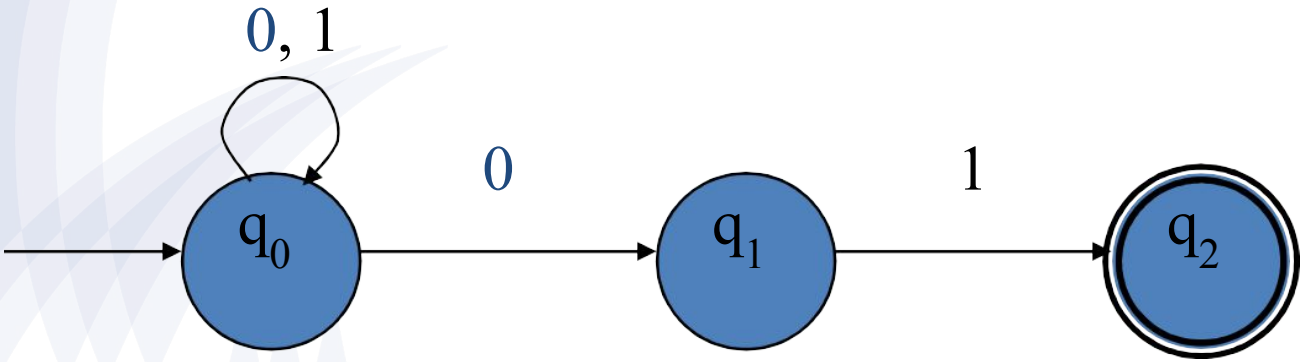
- **Deterministic Finite Automata (DFA)**: At any point when the machine is in a *state with an input symbol, there is a unique next state to go*
- **Non-Deterministic (NFA)**: There can be more than one next state for ***each state-input pair***

Introduction to NFA (II)

Strathmore UNIVERSITY

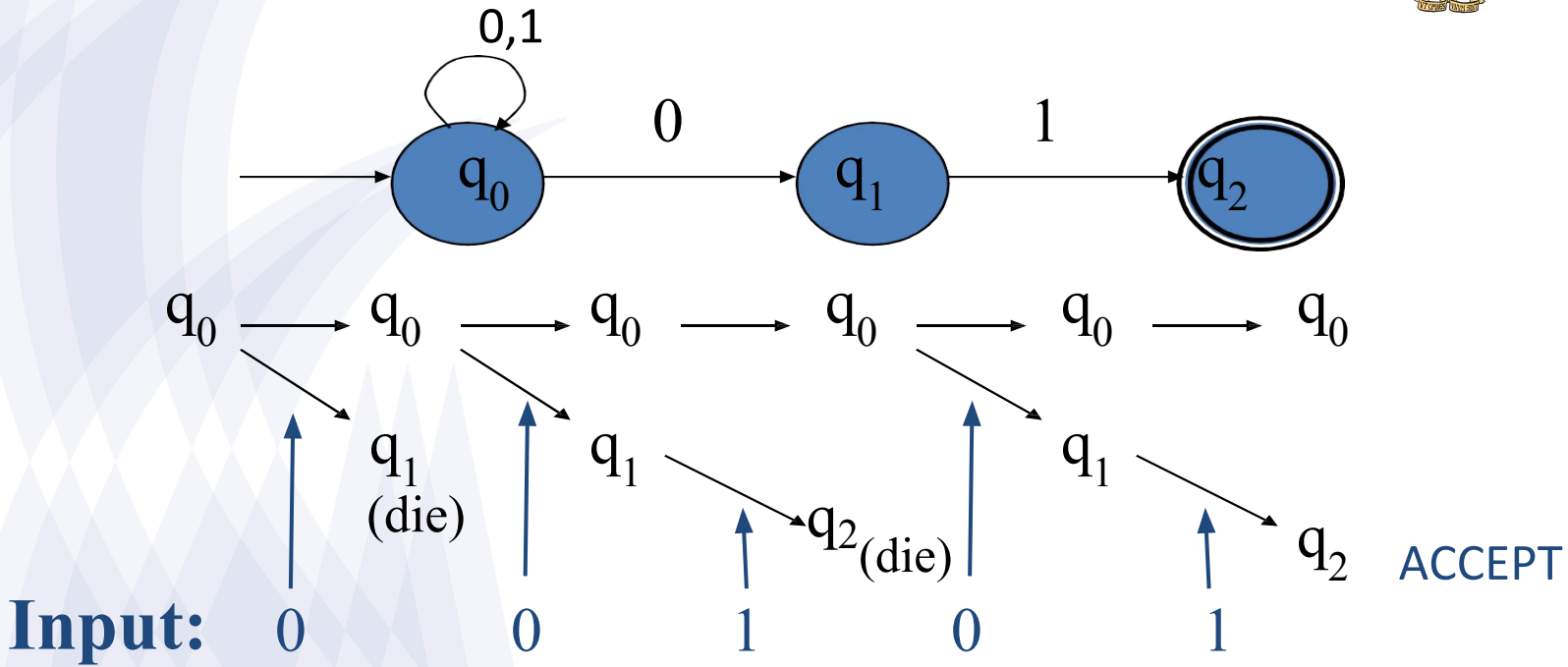
- In NFA, for each state there can be zero, one, two, or more transitions corresponding to a particular symbol
- If NFA gets to state with more than one possible transition corresponding to the input symbol, we say it **branches**
- If NFA gets to a state where there is no Valid Transition, then that branch **dies**
- Example : *NFA accepting all strings ending in 01*

NFA Operation - Example 1



- NFA Machine, M_1

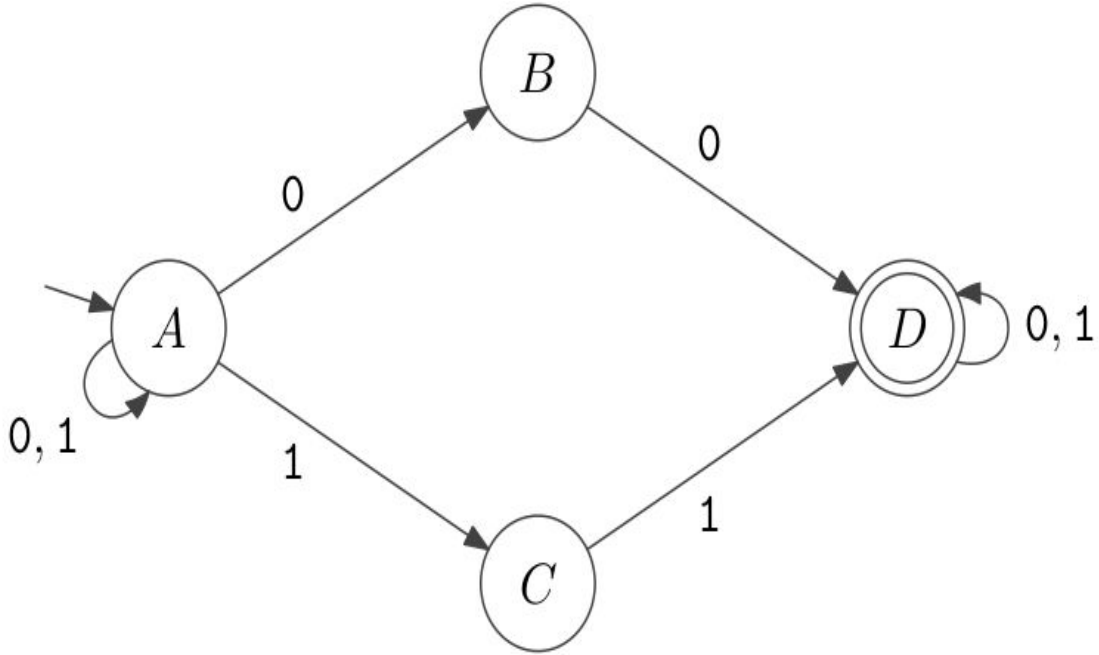
NFA Operation - Example 1(II)



How does this NFA compute? When do NFA accept?

NFA Operation - Example 2

- L = { w|w contains 00 or 11 substring}
- Some sets of strings accepted by this NFA?



NFA Operation (II)

Strathmore UNIVERSITY

- An NFA accepts the input string if there exists some choice of transitions that *leads to ending in an accept state.*
- Thus, one accepting branch is enough for the overall NFA to accept, *but every branch must reject for the overall NFA to reject*

DFA vs NFA

Strathmore UNIVERSITY

- The difference between DFA and NFA are as follows:
  - Every state of a DFA always has exactly one **exiting transition arrow for each symbol** in the alphabet - NFA shown (pg 35), state q_0 has one exiting arrow for 1, but it has **two** for 0
  - In a DFA, labels on the transition arrows are symbols from the alphabet on the other hand, NFA can have a transition on ε

NFA Formal Definition

- Formally, an NFA is a 5-tuple N = (Q, Σ, δ, q_0, F), where all is as DFA, but:

  $\delta: Q \times \Sigma \rightarrow \boldsymbol{P}(Q)$

- $\boldsymbol{P}(Q)$ is a transition function from Q x Σ to the **power set of Q**

NFA Formal Definition (II)

- NFA is defined by a 5 -tuple:

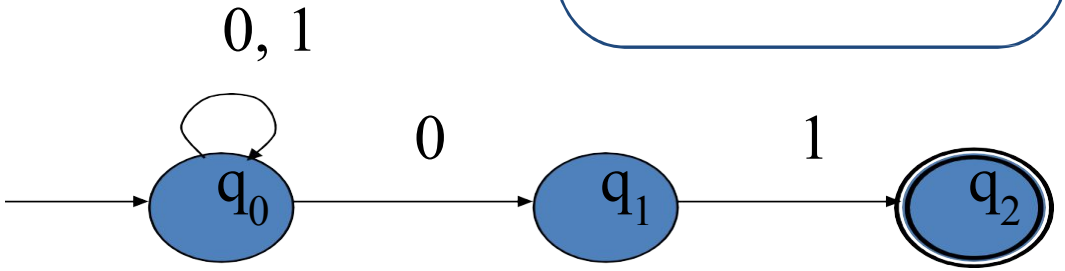
  $(Q, \Sigma, \delta, q_0, F)$

  1. Q: finite set of states
  2. Σ: finite alphabet
  3. **δ: transition function, $\delta : Q \times (\Sigma \cup \varepsilon) \rightarrow \boldsymbol{P}(Q)$**
  4. $q_0 \in Q$: start state
  5. $F \in Q$: set of accept states

NFA **Description** Example

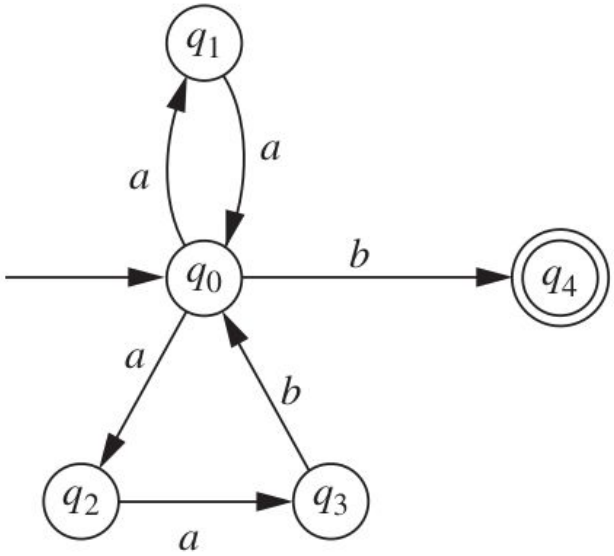
- M = (Q, Σ, δ, q_0, F) where
- – Q = { q_0 , q_1 , q_2 }
  – Σ = {0, 1 }
  – $\delta(q_0, 1) = \{q_0\}$, $\delta(q_0, 0) = \{q_0, q_1\}$
  $\delta(q_1, 1) = \{q_2\}$
  – q_0 is the start state
  – F = $\{q_2\}$

	0	1
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	∅	$\{q_2\}$
q_2	∅	∅



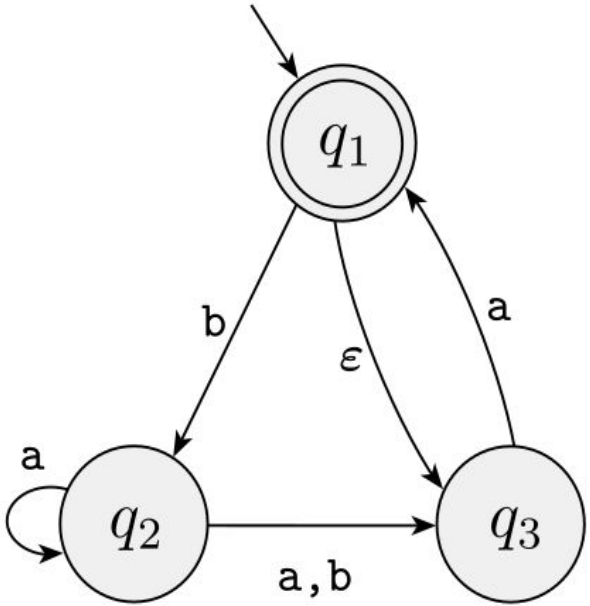
NFA **Description**- Example 2

- Which sets of strings are accepted by the following NFA given Σ = {a,b}
- *Write down the formal description*



NFA **Description** - Example 3

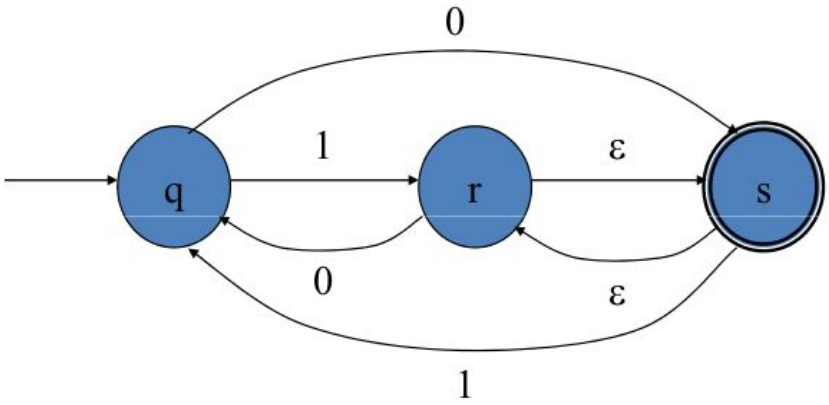
- NFA which accepts *ε*, *a*, *baba*, and *baa*, but not *b*, *bb, and babba*
- *Write down the formal description*



NFA **Description** Definition - Example 4

Strathmore UNIVERSITY

- Other examples of strings accepted by this NFA?
- *Write down the formal description*



"001" is accepted by the path
q→s→r→q→r→s with label 0ε01ε = 001

NFA with ε Transitions

Strathmore UNIVERSITY

- Allows ε to be a label on arcs.
- Transition function δ of an ε-NFA is from

  $Q \times \Sigma \cup \{\varepsilon\} \rightarrow P(Q)$
- Nothing else changes: acceptance of string w is still the existence of a path from the start state to an accept state with label w.
- But ε can appear on arcs, and means the empty string

Strathmore UNIVERSITY

NFA AND DFA EQUIVALENCE

Assignment 01*

Exercise

Strathmore UNIVERSITY

- **The channel selection switch of a television**
  - The channel selection switch of a television set can be turned both clockwise and counterclockwise to select among three possible channels
  - In the first case, channels A, B, and C are selected in the order
  - Model this using a Finite Automaton

Applications of Finite Automata

Strathmore UNIVERSITY

- Finite automata are good models for computers with an extremely limited amount of memory.
- What can a computer do with such a small memory?
- Many useful things!

Applications of Finite Automata (II)

Strathmore UNIVERSITY

- In fact, we interact with such computers all the time, as they lie at the heart of various **electromechanical devices.**
  - The **controller for an automatic door** is one example of such a device
  - Often found at supermarket entrances and exits, automatic doors swing open when sensing that a person is approaching

Applications of Finite Automata (III)

Strathmore UNIVERSITY

- In fact, we interact with such computers all the time, as they lie at the heart of various **electromechanical devices.**
  - An automatic door has a pad in front to detect the presence of a person about to walk through the doorway

Applications of Finite Automata (IV)

Strathmore UNIVERSITY

- In fact, we interact with such computers all the time, as they lie at the heart of various **electromechanical devices.**
  - Another pad is located to the rear of the doorway so that the controller can hold the door open long enough for the person to pass all the way through and also so that the door does **not strike someone standing behind** it as it opens

Applications of Finite Automata (V)

Strathmore UNIVERSITY

- Other applications of Finite Automata include:
  - Software for designing and checking the behaviour of digital circuits
  - The “lexical analyzer” of a typical compiler, i.e. the compiler component that breaks the input text into logical units, such as **identifiers,** keywords, and punctuation

Applications of Finite Automata (VI)

Strathmore UNIVERSITY

- Other applications of Finite Automata include:
  - Software for scanning large bodies of text, such as collections of web pages, to find occurrences of words, phrases, or other patterns.
  - Software for verifying systems of all types that have a finite number of distinct states, such as communication protocols or protocols for secure exchange of information

LIMITATIONS OF FINITE STATE MACHINES (FSM)

Strathmore UNIVERSITY

- The fundamental limitation of FSMs is that they have no capacity for remembering arbitrarily large amounts of information.
- Any computation that requires the memory of ***arbitrarily long sequences*** cannot be carried out on a FSM

Strathmore
UNIVERSITY

Questions