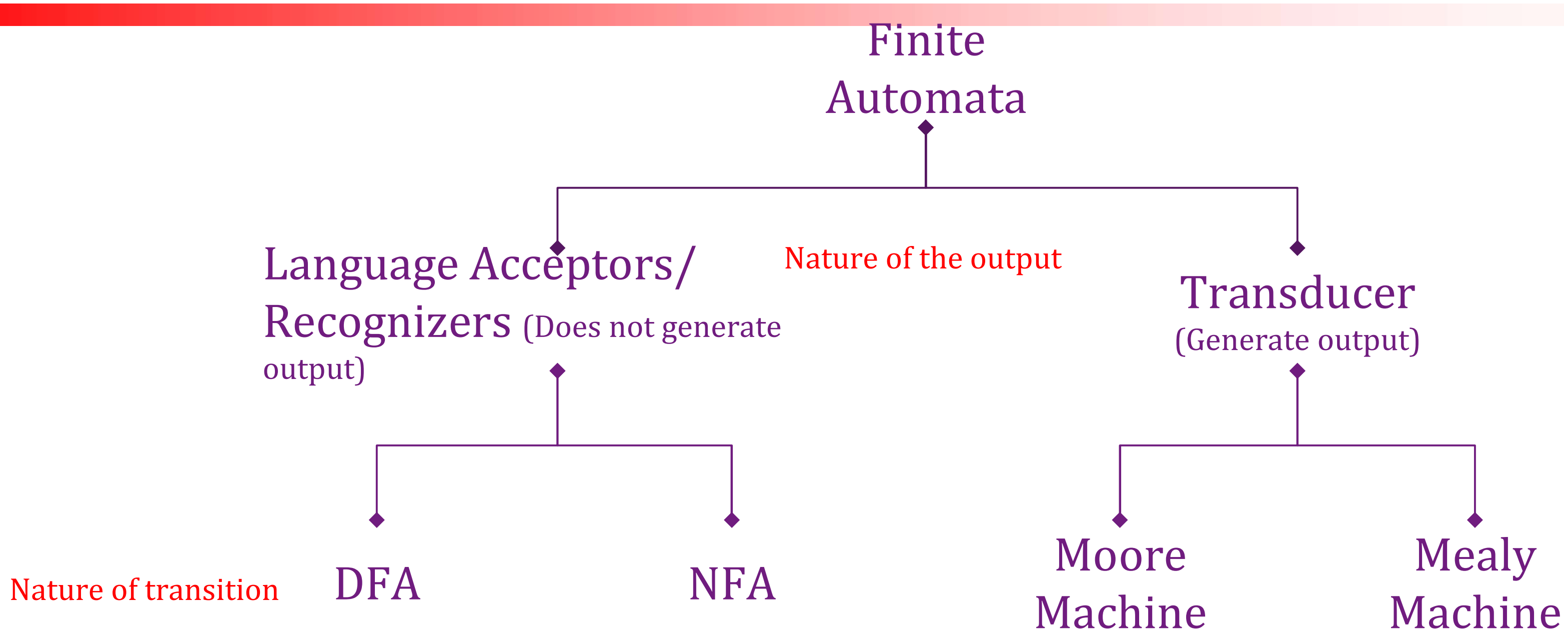
SCS 2112 : Automata Theory

Dinuni Fernando, PhD

UCSC

3. NFA

Classification of FSMs



- Moore machines : output depends only on the state.
- Mealy machines : output depends on input and state.

Transducers

Definition : A transducer is defined by the tuple M = (Q, Σ, Γ, δ, q_0, ω) where

– Q : is a finite set of **internal states**

– Σ : is a finite set of symbols called the **input alphabet**

– Γ : is the output alphabet

– $\delta : Q \times \Sigma \longrightarrow Q$ is a function called the **transition function**

– $q_0 \in Q$ is the **initial state**

– ω : output function

UCSC

Transducers

Transducers

- **Mealy model**
  $\omega : Q \times \Sigma \longrightarrow \Gamma$
  Output depends on input and state
- **Moore Model**
  $\omega : Q \longrightarrow \Gamma$
  Output depends only on the state

Mealy Machine

- Is a finite state machine whose output depends on the present state and the present input
- Described using a 6 tuple $(Q, \sum, \Gamma, \delta, q_0, \omega)$, where

  – Q : is a finite set of **internal states**

  – $\sum$: is a finite set of symbols called the **input alphabet**

  – Γ : is the output alphabet

  – $\delta : Q \times \sum \longrightarrow Q$ is a function called the **input transition function**

  – $q_0 \in Q$ is the **initial state**

  – $\omega : Q \times \sum \longrightarrow \Gamma$ is a output transition function

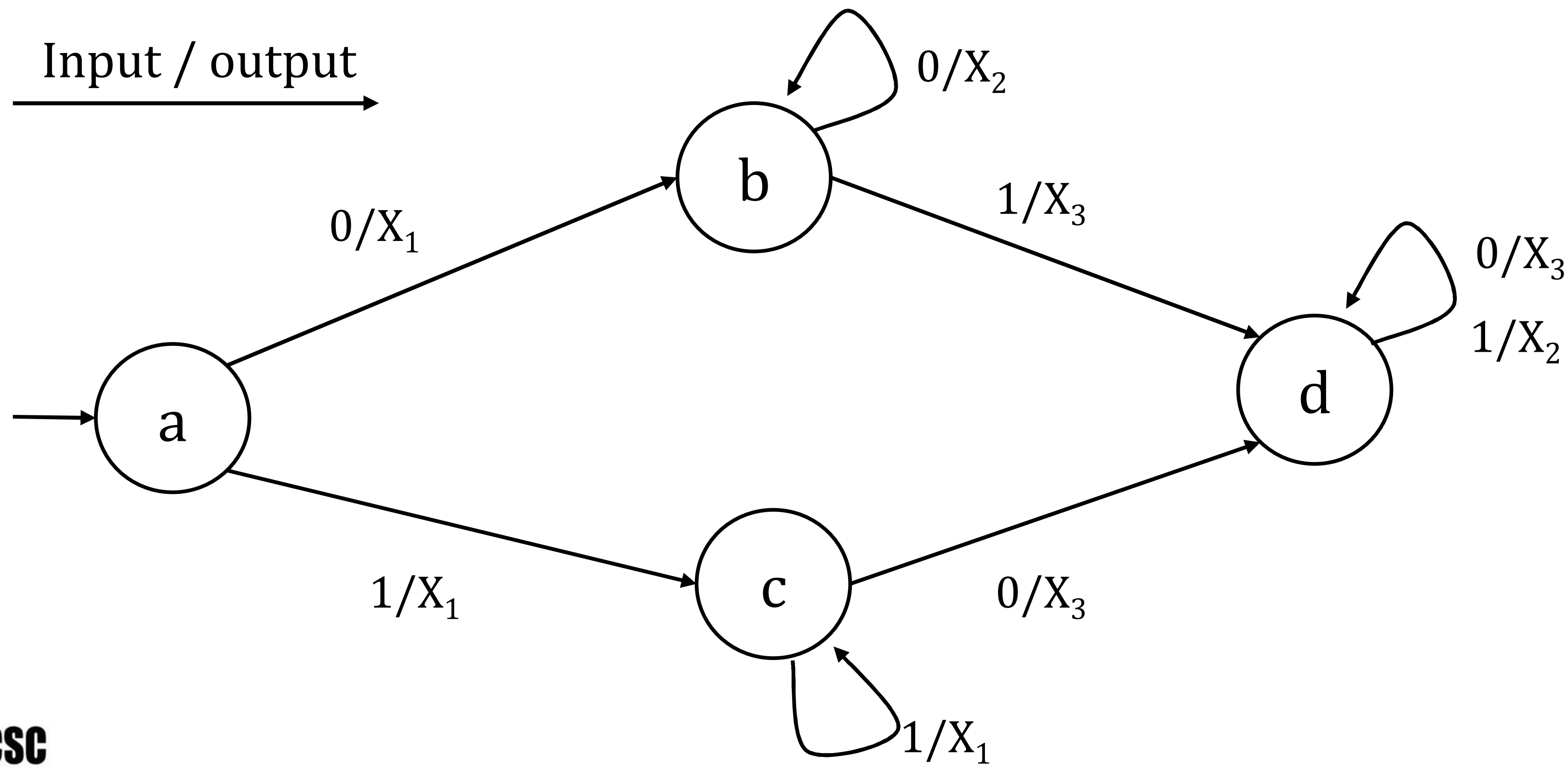
UCSC

State transition table of Mealy machine

Present state	Next State			
	Input = 0		Input = 1	
	State	Output	State	Output
⟶a	b	X_1	c	X_1
b	b	X_2	d	X_3
c	d	X_3	c	X_1
d	d	X_3	d	X_2

UCSC

State transition diagram for Mealy machine



Moore Machine

- Is a finite state machine whose output only depends on the present state
- Described using a 6 tuple $(Q, \sum, \Gamma, \delta, q_0, \omega)$, where

  – Q : is a finite set of **internal states**

  – $\sum$: is a finite set of symbols called the **input alphabet**

  – Γ : is the output alphabet

  – $\delta : Q \times \sum \longrightarrow Q$ is a function called the **input transition function**

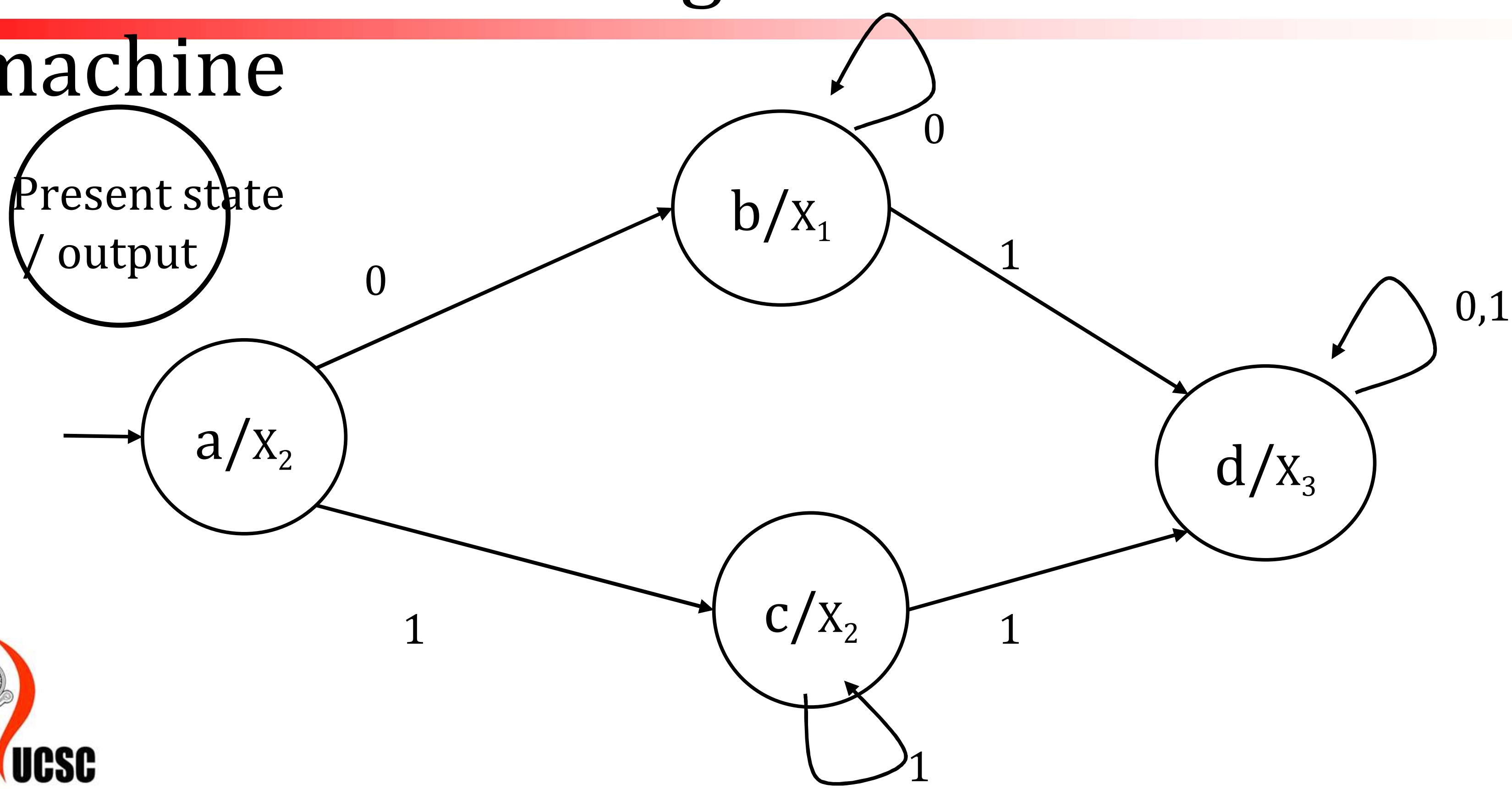
  – $q_0 \in Q$ is the **initial state**

  – $\omega : Q \longrightarrow \Gamma$ is a **output transition function**

State transition table of Moore machine

Present state	Next State		Output
	Input = 0	Input = 1	
→a	b	c	X_2
b	b	d	X_1
c	c	d	X_2
d	d	d	X_3

State transition diagram for Moore machine



Conversion of Mealy machine to Moore Machine

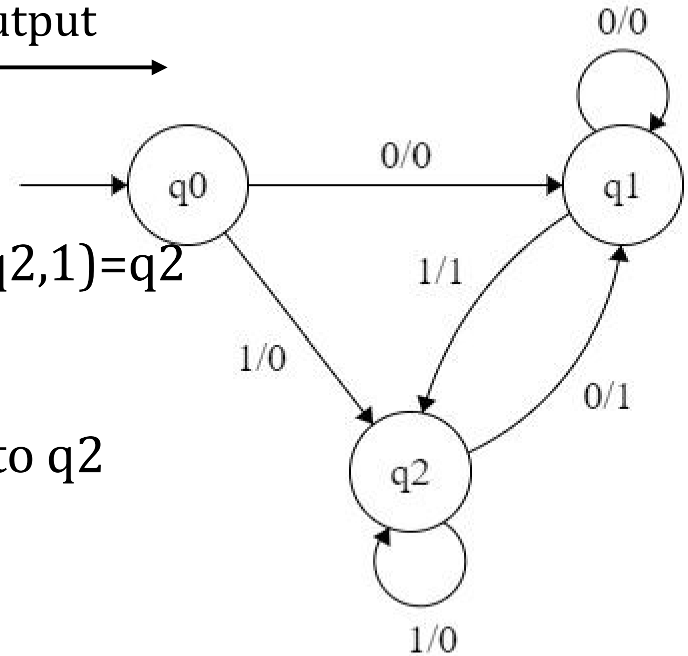
Input/output

Input : 11

Transition : δ(q0,11) = δ(δ(q0,1),1)= δ(q2,1)=q2

Output : 00

(q0 to q2 transition has Output 0 and q2 to q2 transition also has Output 0)



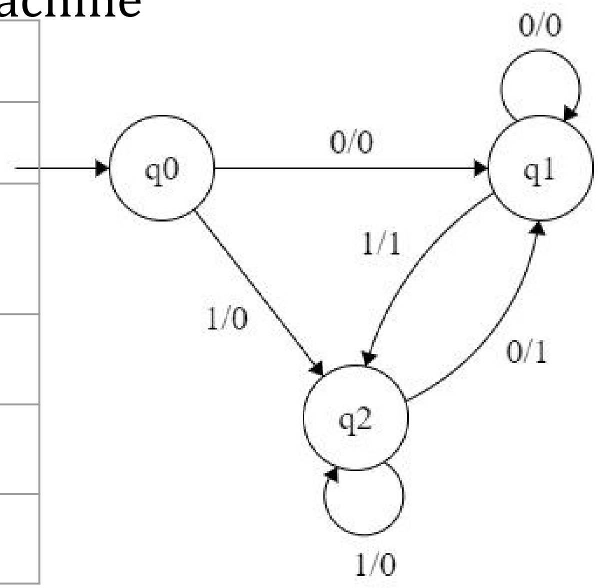
Mealy Machine

Conversion of Mealy machine to Moore Machine

Input – Mealy Machine **Output** – Moor Machine

Step 1 : First consider, transition table of mealy machine

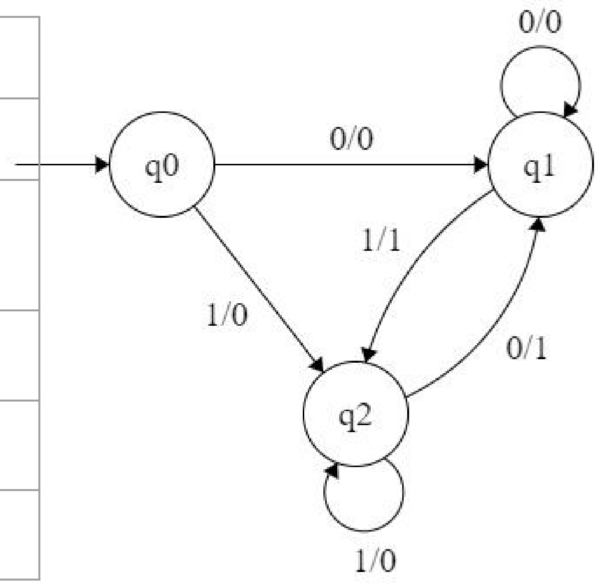
Present state	Next State			
	Input = 0		Input = 1	
	State	Output	State	Output
⟶q0	q1	0	q2	0
q1	q1	0	q2	1
q2	q1	1	q2	0



Conversion of Mealy machine to Moore Machine

Step 2 : First find out states which have more than 1 output associated with them. Such states are q1 and q2.

Present state	Next State			
	Input = 0		Input = 1	
	State	Output	State	Output
→q0	q1	0	q2	0
q1	q1	0	q2	1
q2	q1	1	q2	0



Conversion of Mealy machine to Moore Machine

Step 3 : Create 2 states for these states. For q1, two states will be q10 (with output 0) and q11 (with output 1). Similarly, q2's new states are q20, q21.

Step 4 : Create an empty Moore machine with new generated states. For Moore machine, output will be associated to each state irrespective of its input.

Present state	**Next State**		**Output**
	Input = 0	**Input = 1**	
→q0			
q10			
q11			
q20			
q21			

UCSC

Conversion of Mealy machine to Moore Machine

Step 5 : Fill the entries of next state using mealy machine transition table.

For q0 on input 0 next state is q10 (q1 with output 0). For q1 (for both q10 and q11) on input 0, next state is q10. for q10 and q11 on input 1, next state is q21. Similarly you will fill the entire table.

Present state	**Next State**		**Output**
	Input = 0	**Input = 1**	
→q0	q10	q20	0
q10	q10	q21	0
q11	q10	q21	1
q20	q11	q20	0
q21	q11	q20	1

Conversion of Moore machine to Mealy Machine

Consider the Moore machine transition table.

Present state	Next State		Output
	Input = 0	Input = 1	
→q0	q10	q20	0
q10	q10	q21	0
q11	q10	q21	1
q20	q11	q20	0
q21	q11	q20	1

Conversion of Moore machine to Mealy Machine

Step1 : Construct an empty mealy machine using all states of Moore machine.

Present state	Next State			
	Input = 0		Input = 1	
	State	Output	State	Output
⟶q0				
q10				
q11				
q20				
q21				

Conversion of Moore machine to Mealy Machine

Step2 : Next state of each state can be directly found from Moore machine transition table.

Present state	**Next State**			
	Input = 0		**Input = 1**	
	State	**Output**	**State**	**Output**
⟶q0	q10		q20	
q10	q10		q21	
q11	q10		q21	
q20	q11		q20	
q21	q11		q20	

Conversion of Moore machine to Mealy Machine

Step3 : As we can see output corresponding to each input in Moore machine transition table. By using this, we can fill output entries, as follows, q10,q11,q20 and q21 are 0,1,0,1 respectively.

Present state	**Next State**			
	Input = 0		**Input = 1**	
	State	**Output**	**State**	**Output**
⟶q0	q10	0	q20	0
q10	q10	0	q21	1
q11	q10	0	q21	1
q20	q11	1	q20	0
q21	q11	1	q20	0

Conversion of Moore machine to Mealy Machine

Step4 : According to the table q10 and q11 are similar to each other (same value of next state and output for different inputs). Similarly q20 and q21 are similar. So, we can eliminate q11 and q21.

Present state	Next State			
	Input = 0		Input = 1	
	State	Output	State	Output
⟶q0	q10	0	q20	0
q10	q10	0	q21	1
q11	q10	0	q21	1
q20	q11	1	q20	0
q21	q11	1	q20	0

UCSC

Conversion of Moore machine to Mealy Machine

Step5 : After eliminating q11 and q21.

Present state	Next State			
	Input = 0		Input = 1	
	State	Output	State	Output
⟶q0	q10	0	q20	0
q10	q10	0	q21	1
q20	q11	1	q20	0

Non-deterministic Finite Automata (NFA)

Definition : A non-deterministic finite automata (**NFA**) is defined by the tuple $M = (Q, \Sigma, \delta, q_0, F)$ where Q, Σ, q_0, F are defined as for dfa, but δ is defined as below

$\delta : Q \times (\Sigma \cup \{\in\}) \longrightarrow 2^Q$

Where 2^Q is the powerset of Q

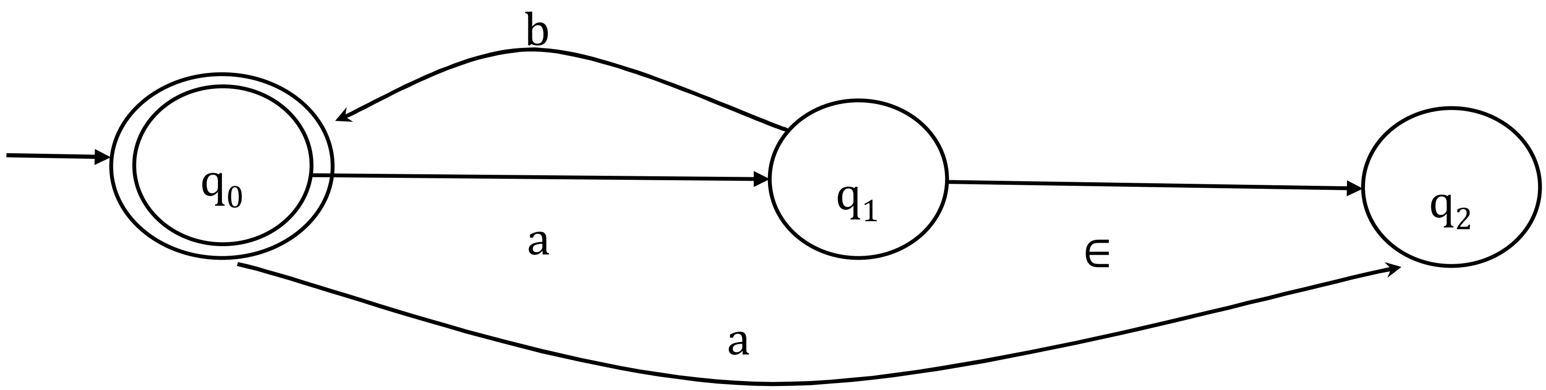
δ is a relation (not a function)

Differences between DFA and NFA

- The range of δ is a set (i.e, powerset).
  - There may be multiple transitions defined on the same state on the same input.
  - State is not a single element of Q but a subset of it.
- ∈ is allowed as the second argument of δ
  - NFA can make a transition without consuming an input symbol.
  - Input mechanism is stationary on some moves

$\{\} \in 2^Q$ – transitions may not have been defined for some inputs

Examples of a NFA

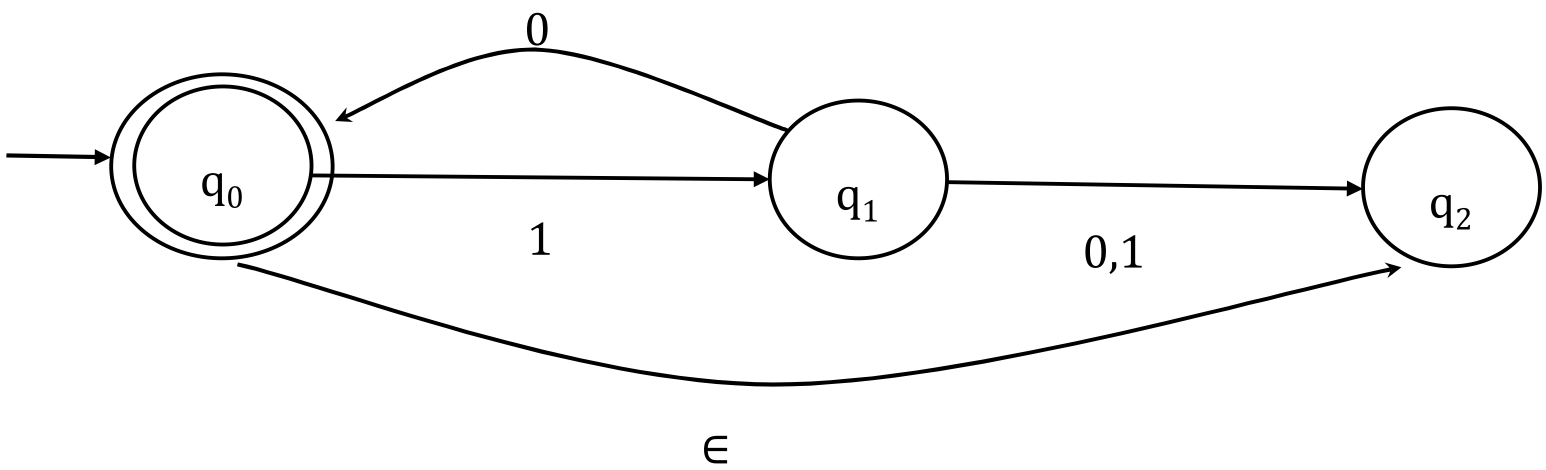


- Several edges with the same label originate from one vertex.
- Has ∈ transitions
- Some transitions are unspecified eg. $\delta(q_2,a)$

Transition table for the NFA

State	Input symbol		
	a	b	∈
q_0	$\{q_1, q_2\}$	--	--
q_1	--	$\{q_0\}$	$\{q_2\}$
q_2	--	--	--

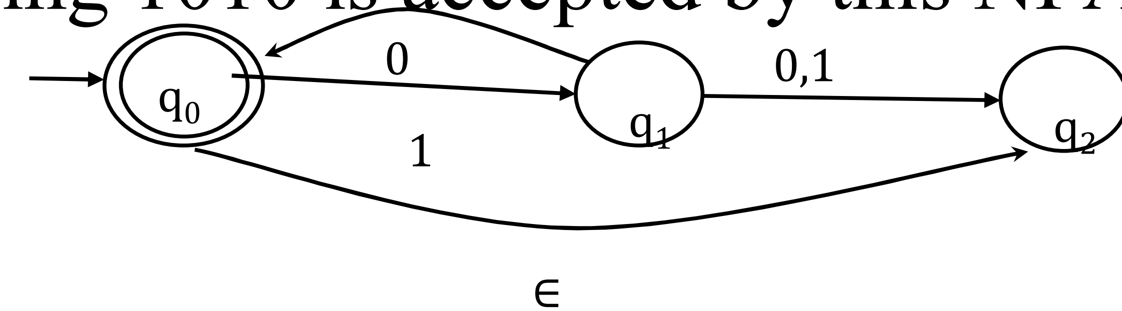
Example 1



- Is the string 1010 is accepted by this NFA?

Example 1

- Is the string 1010 is accepted by this NFA?



$$q_0 \xrightarrow{1} q_1 \xrightarrow{0} q_0 \xrightarrow{1} q_1 \xrightarrow{0} q_0$$ Accepting state

$q_0 \xrightarrow{\in} q_2$ stuck

$q_1 \xrightarrow{0} q_2$ stuck