Finite Automata

Mohammad Hasan
CSE, CUET

Finite Automata

- A finite automaton has a set of states and it control moves from state to state in response to external inputs
- Deterministic: automation cannot be in more than one state at any time
- Nondeterministic: automation can be several states at once

Informal Picture of Finite Automata

- Finite automata are good models for computers with an extremely limited amount of memory

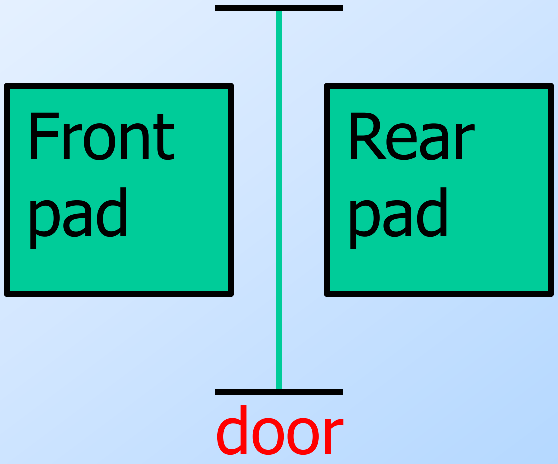
-electromechanical devices

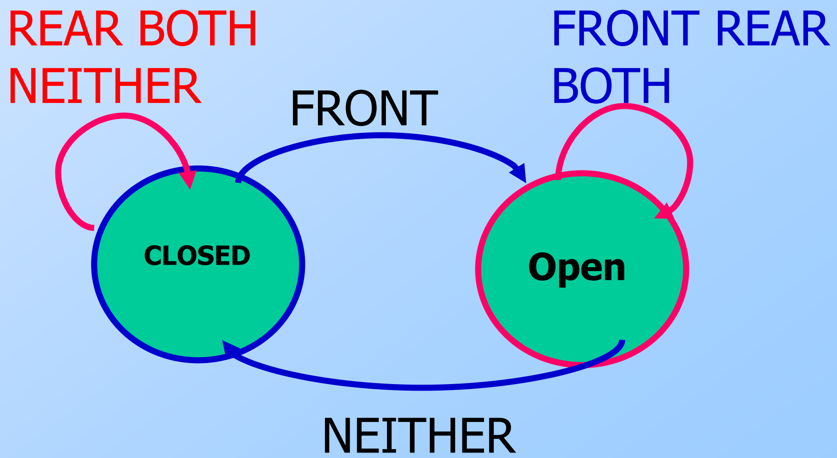
- The controller for an automatic door is one example of such a device
- Often found in supermarket entrances/exits, automatic doors swing open when sensing that a person is approaching

Informal Picture of Finite Automata

- Two states: OPEN or CLOSED
- 04 input conditions:

1. FRONT (a person is standing on the pad in front of the doorway)
2. REAR (a person is standing on the pad to the rear of the doorway)
3. BOTH (people are standing on both pads)
4. NEITHER (no one is standing on either pad)





State table

Input signal				
	NEITHER	FRONT	REAR	BOTH
CLOSED	CLOSED	OPEN	CLOSED	CLOSED
OPEN	CLOSED	OPEN	OPEN	OPEN

- Controller moves from state to state, depending on input
- when CLOSED state & input NEITHER/REAR, it remains in CLOSED state
- If the input BOTH, it stays CLOSED because opening the door Risks knocking someone over on the rear pad
- But if input FRONT, it moves to the OPEN state
- In the OPEN state, if input FRONT/REAR/BOTH, it remains OPEN

Other Examples

- This controller is a computer that has just a single bit of memory, capable or recording which is two states the controller is in
- Others: elevator controller, dishwashers, thermostat, digital watches, calculators

etc

Deterministic Finite Automata (DFA)

- **Deterministic**: on each input there is one state to which the automaton can transition from its current state
- In terms of 05 tuples:

$$\boxed{A = (Q, \sum, \delta, q_0, F)}$$

- **A**: DFA, **Q**: set of states, **Σ**: input symbols, **δ**: transition function, **q_0**: start state, **F**: set of accepting state

DFA

- A formalism for defining languages, consisting of:
  1. A finite set of *states* (Q, typically).
  2. An *input alphabet* (Σ, typically).
  3. A *transition function* (δ, typically).
  4. A *start state* (q_0, in Q, typically).
  5. A set of *final states* ($F \subseteq Q$, typically).
     - "Final" and "accepting" are synonyms denoted by Double Loops.

The Transition Function

- Takes two arguments: a state & an input symbol; returns a state
- δ represented by an arc between states and the labels on the arcs.
- $\delta(q, a)$ = the state that the DFA goes to when it is in state *q* & input *a* is received.
- q state (current) to p state (next): an arc labeled with a

How a DFA Process Strings

- How the DFA decides whether or not to 'accept' a sequence of input symbols
- The language of the DFA is the set of all strings that the DFA accepts.
- Sequence of input symbols: $a_1, a_2, \ldots\ldots a_n$
- initial state: q_0
- Transition function, $\delta(q_0, a_1) = q_1$ to find state that the DFA A enters after processing the first input symbol a_1

How a DFA Process Strings

- Process next input symbol, a_2: $\delta(q_1, a_2)= q_2$
- continue in this way, finding states q_3, q_4,.. q_n, such that $\delta(q_{i-1}, a_i)= q_i$ for each i
- If q_n is a member of F, then the input $a_1a_2\ldots a_n$ is accepted,
- if not then it is rejected.

Example

- Specify a DFA that accepts all and only the strings of 0's and 1's that have the sequence 01 somewhere in the string.
- L as:

{w| w is of the form x01y for some strings

x & y consisting of 0's & 1's only}

- {x01y| x & y are any strings of 0's & 1's}

Example

- Strings: 01, 11010, 100011 exists in L
- ε, 0, & 111000 not exist in L
- $\Sigma=\{0,1\}$
- $\delta(q_0,1)=q_0$
- $\delta(q_0,0)=q_2$
- $\delta(q_2,0)=q_2$
- $\delta(q_2,1)=q_1$

Example

- $\delta(q_1,0)=\delta(q_1,1)=q_1$
- $Q = \{q_0, q_1, q_2\}$, $F = \{q_1\}$
- The complete specification of the automation A that accepts the L of strings that have a 01 substring:

  $A = (\{q_0, q_1, q_2\}, (0,1), \delta, q_0, \{q_1\})$

Notations for DFA’s

- Specifying a DFA as 5tuple with a detailed description of the δ transition function: tedious & hard to read
- Two notations:

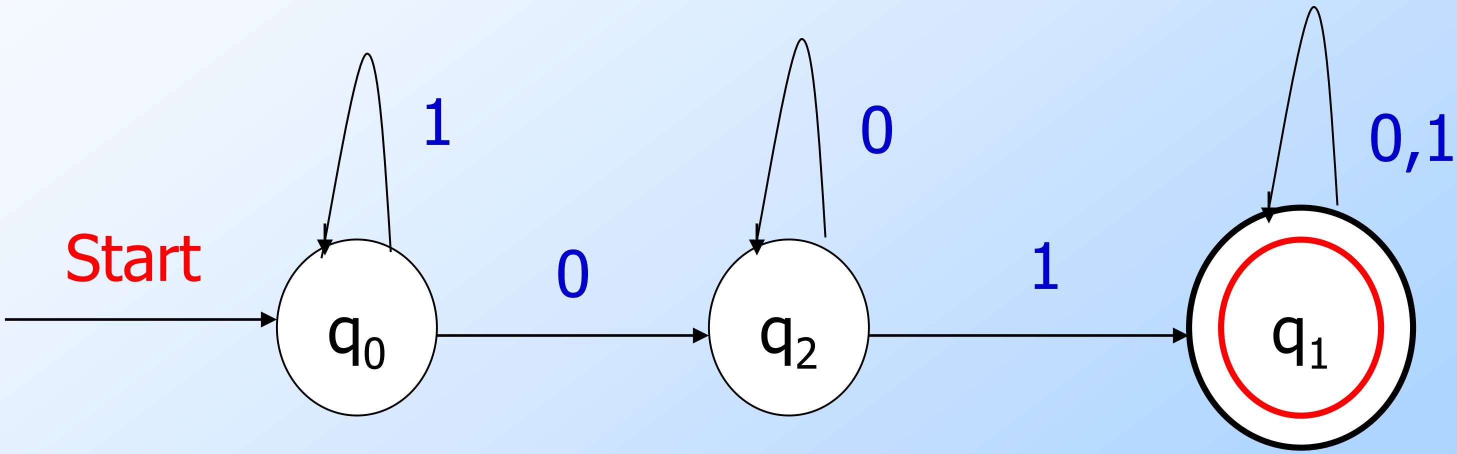
1. Transition diagram
2. Transition table

Transition Diagram

- 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.

Example

- **Draw the Transition Diagram for the DFA accepting all string with a substring 01.**

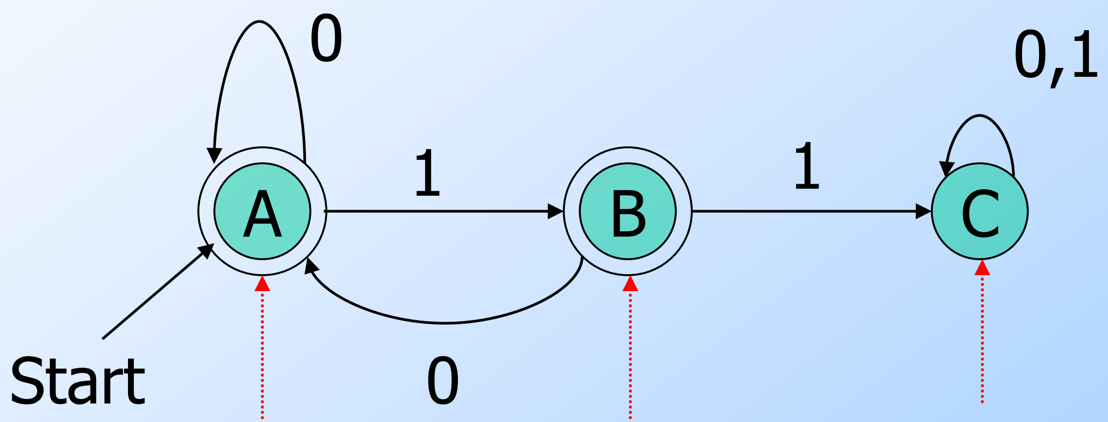


$$A=(\{q_0,q_1,q_2\},\{0,1\}, \delta ,q_0,\{q_1\})$$

Check with the string 01,11010,100011, 0111,110101,11101101, 111000

Another Example

Accepts all strings without two consecutive 1's.



Previous string OK, does not end in 1.

Previous String OK, ends in a single 1.

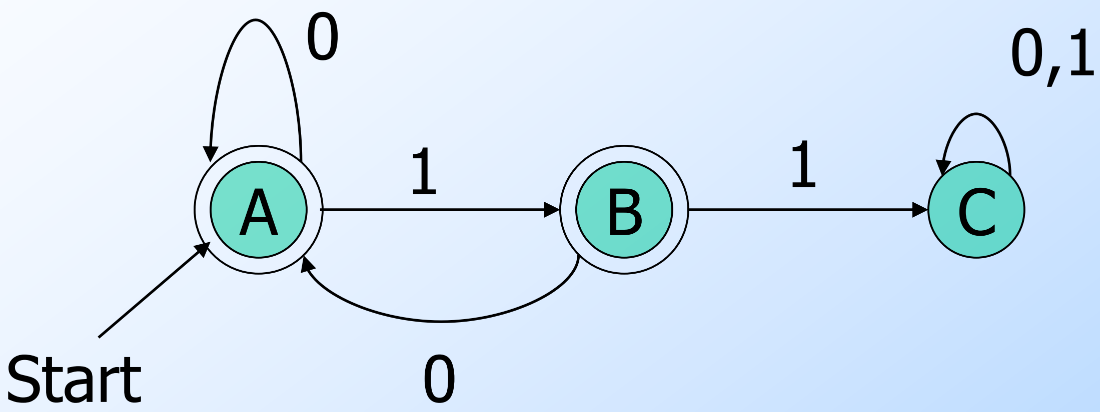
Consecutive 1's have been seen.

Transition Table

- a conventional, tabular representation of a function like δ that takes two arguments & return a value.
  - Rows: states; **corresponding to state q**
  - Columns: inputs; **corresponding to input a is the state δ(q,a)**

- **The start state is marked with an arrow and accepting states are marked with a star.**

Example



Final states starred

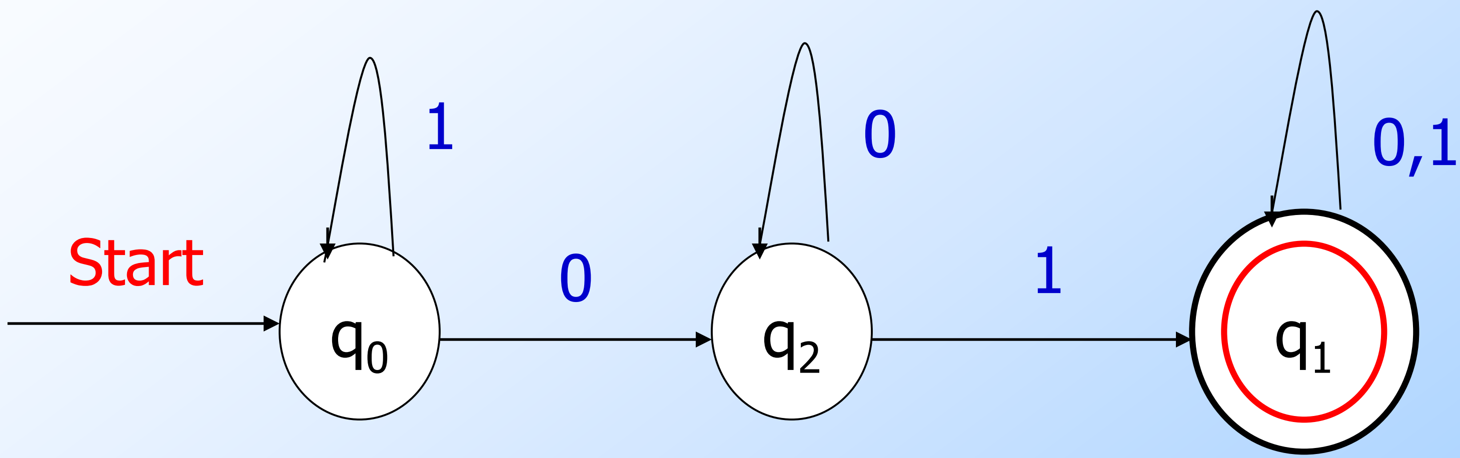
Columns = input symbols

Arrow for start state

	0	1
→ * A	A	B
* B	A	C
C	C	C

Rows = states

Example



- $(q_0,0)=q_2$
- $(q_0,1)=q_0$
- $(q_1,0)=q_1$
- $(q_1,1)=q_1$
- $(q_2,0)=q_2$
- $(q_2,1)=q_1$

	0	1
$\rightarrow q_0$	q_2	q_0
$*q_1$	q_1	q_1
q_2	q_2	q_1

Extended Transition Function

- An Extended Transition Function that describes what happen when we start in any state and follow any sequence of inputs.

δ → Transition Function

$\hat{\delta}$ → Extended Transition Function

- a function that takes a state q & a string w & returns a state p-the state that the automaton reaches when starting in state q & processing the sequence of inputs w

Extended Transition Function

- We describe the effect of a string of inputs on a DFA by extending δ to a state and a string.
- Induction on length of string.
- Basis: $\hat{\delta}(q, \epsilon) = q$
- Induction: $\hat{\delta}(q,wa) = \delta(\hat{\delta}(q,w),a)$
  - w is a string; a is an input symbol.

Induction

2. $\hat{\delta}(q,w)=\delta(\hat{\delta}(q,x),a)=\delta(p,a)$

   w →xa

   a →last symbol = 1

   w →string = 1101

   x →string consisting of all but the last symbol = 110

Extended δ: Intuition

- Convention:
  - … w, x, y, x are strings.
  - a, b, c,… are single symbols.
- Extended δ is computed for state q and inputs $a_1a_2…a_n$ by following a path in the transition graph, starting at q and selecting the arcs with labels a_1, a_2,…,a_n in turn.

Example: Extended Delta

	0	1
A	A	B
B	A	C
C	C	C

$\delta(B,011) = \delta(\delta(B,01),1) = \delta(\delta(\delta(B,0),1),1) =$

$\delta(\delta(A,1),1) = \delta(B,1) = C$

Delta-hat

- In book, the extended δ has a "hat" to distinguish it from δ itself.
- Not needed, because both agree when the string is a single symbol.
- $\hat{\delta}(q, a) = \delta(\hat{\delta}(q, \epsilon), a) = \delta(q, a)$

Extended deltas

Example

- Let us design a DFA to accept the language

  L={w | w has both an even number of 0's and even number of 1's}

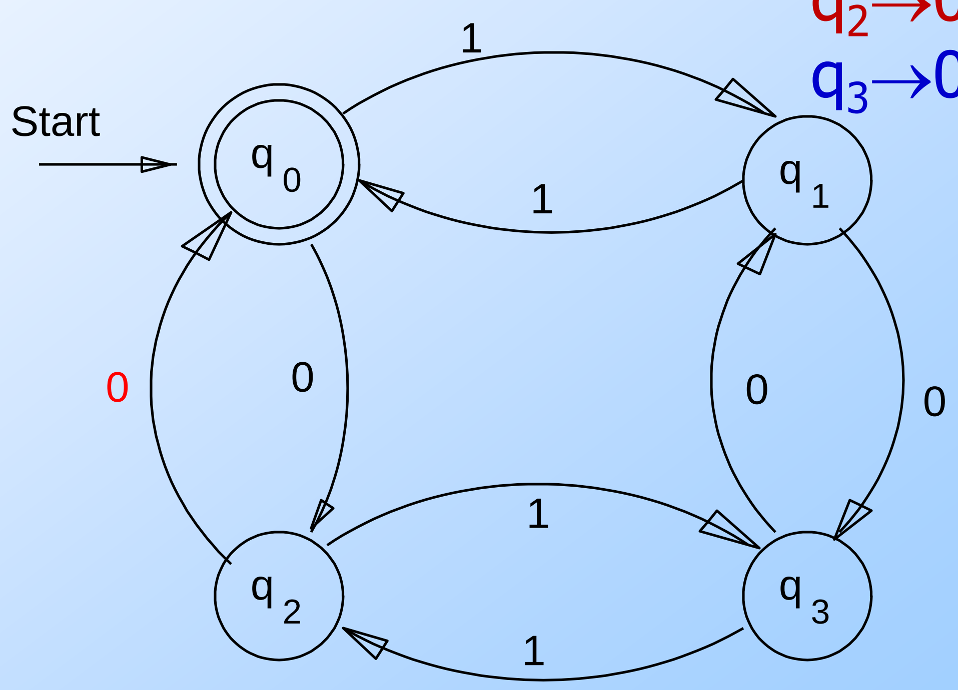
$q_0 \rightarrow 0(\text{even})\ 1\ (\text{even})$

$q_1 \rightarrow 0(\text{even})\ 1\ (\text{odd})$

$q_2 \rightarrow 0(\text{odd})\ 1\ (\text{even})$

$q_3 \rightarrow 0(\text{odd})\ 1\ (\text{odd})$

	0	**1**
$\rightarrow *q_0$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2



Example with Transition Table

	0	1
$\rightarrow * q_0$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_3	q_2

- $\hat{\delta}(q_0, \epsilon) = q_0$
- $\hat{\delta}(q_0,1) = \delta(\hat{\delta}(q_0, \epsilon),1) = \delta(q_0,1) = q_1$
- $\hat{\delta}(q_0,11) = \delta(\hat{\delta}(q_0, 1),1) = \delta(q_0,1) = q_0$
- $\hat{\delta}(q_0,110) = \delta(\hat{\delta}(q_0,11),0) = \delta(q_0,0) = q_2$
- $\hat{\delta}(q_0,1101) = \delta(\hat{\delta}(q_0,110),1) = \delta(q_2,1) = q_3$
- $\hat{\delta}(q_0,11010) = \delta(\hat{\delta}(q_0,1101),0) = \delta(q_3,0) = q_1$
- $\hat{\delta}(q_0,110101) = \delta(\hat{\delta}(q_0,11010),1) = \delta(q_1,1) = q_0$

- **w= 110101**
- **x= 11010**
- **a= 1**

$\hat{\delta}(q_0,110101)=q_0$

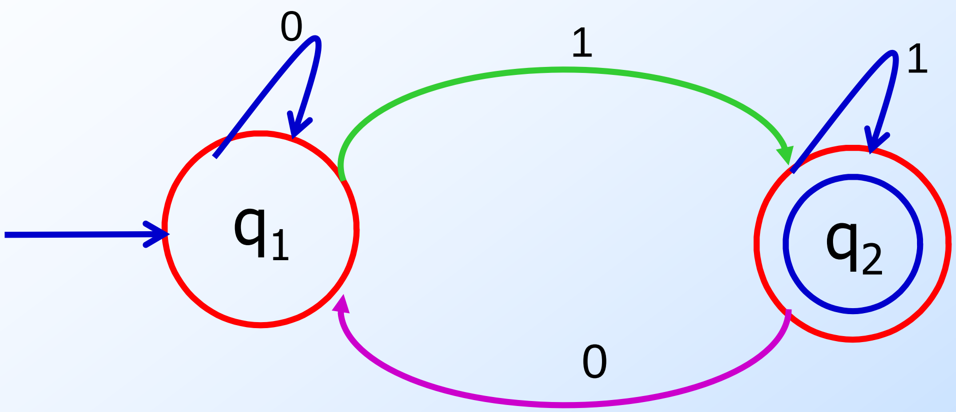
Accepted

Example: Try Yourself

- A = {w | w contains at least one 1 and an even number of 0s follow the last 1
- **Hints:** $A_1 = (Q, \Sigma, \delta, q_1, F)$

1. $Q = \{q_1, q_2, q_3\}$
2. $\Sigma = \{0, 1\}$
3. δ try yourself
4. Start state: q_1
5. Final state: $\{q_2\}$

Example

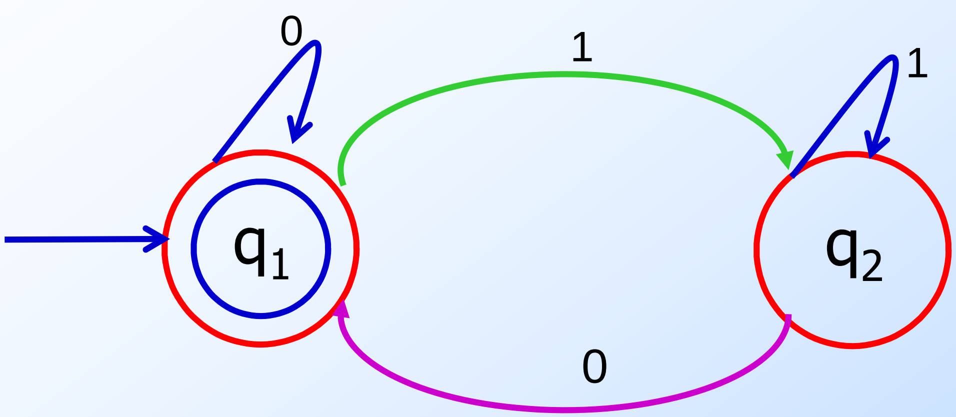


- $A_2 = (\{q_1, q_2\}, (0,1), \delta, q_1, \{q_2\})$
- Transition function, δ

Try: 1101, 11010, 0011010
L(A2) = {w | w ends in a 1}

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

Example



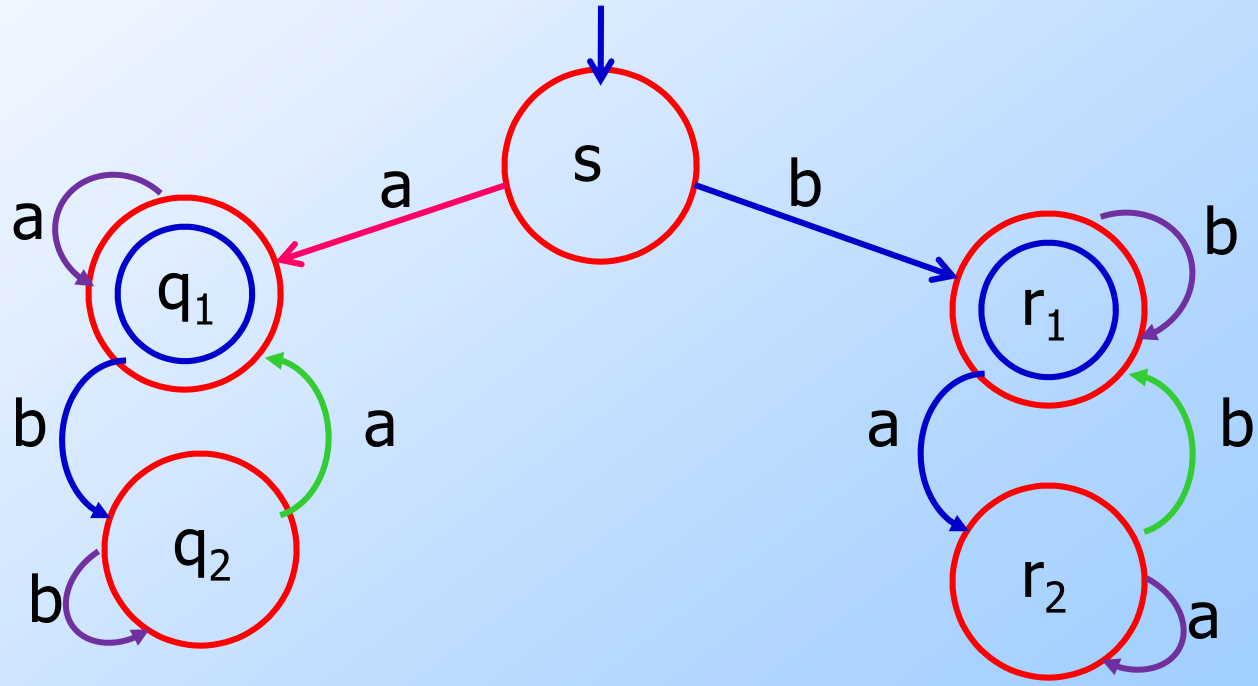
- $A_3 = (\{q_1, q_2\}, (0,1), \delta, q_1, \{q_1\})$
- Transition function, δ

	0	1
$\rightarrow{}^*q_1$	q_1	q_2
q_2	q_1	q_2

Try: 1101, 11010, 0011010
L(A_3) = {w | w is ε or ends in a 0}

Example

- A_4 accepts all strings that start and end with a or that start and end with b
- A_4 accepts strings, that start and end with the same symbol

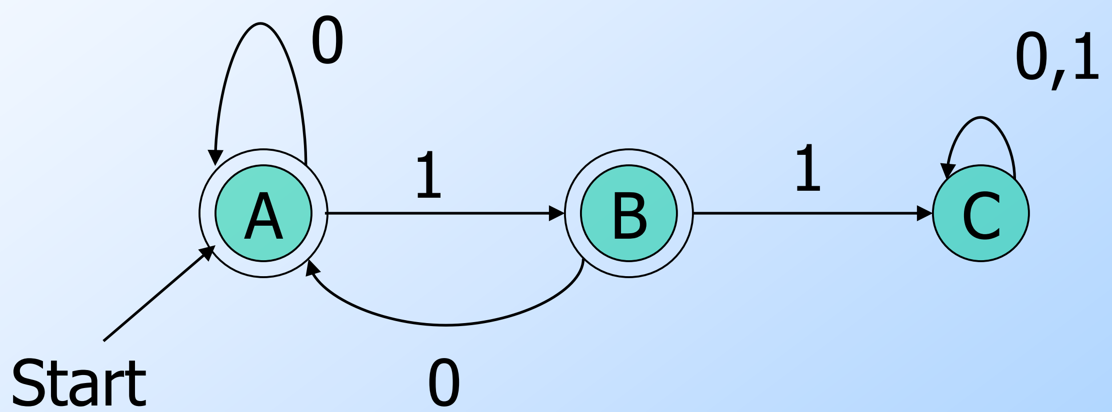


Language of a DFA

- Automata of all kinds define languages.
- If A is an automaton, L(A) is its language.
- For a DFA A, L(A) is the set of strings labeling paths from the start state to a final state.
- Formally: L(A) = the set of strings w such that $\delta(q_0, w)$ is in F.

Example: String in a Language

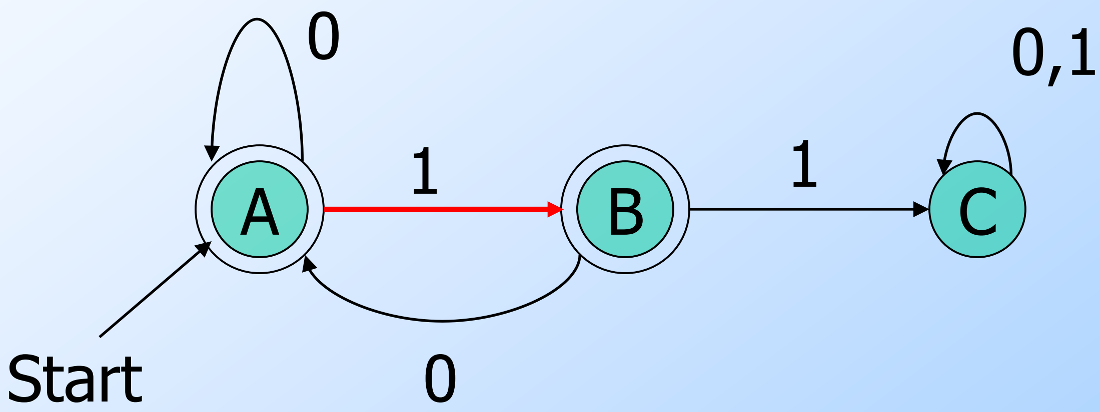
String 101 is in the language of the DFA below.
Start at A.



Example: String in a Language

String 101 is in the language of the DFA below.

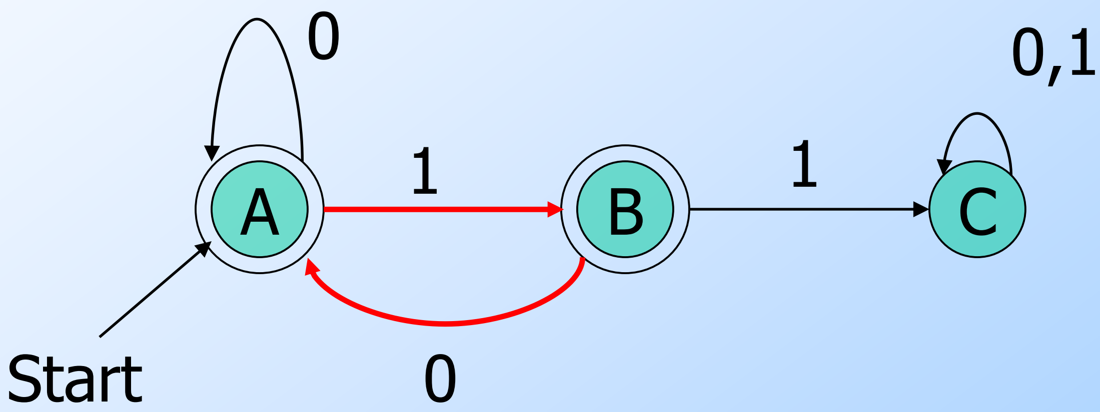
Follow arc labeled 1.



Example: String in a Language

String 101 is in the language of the DFA below.

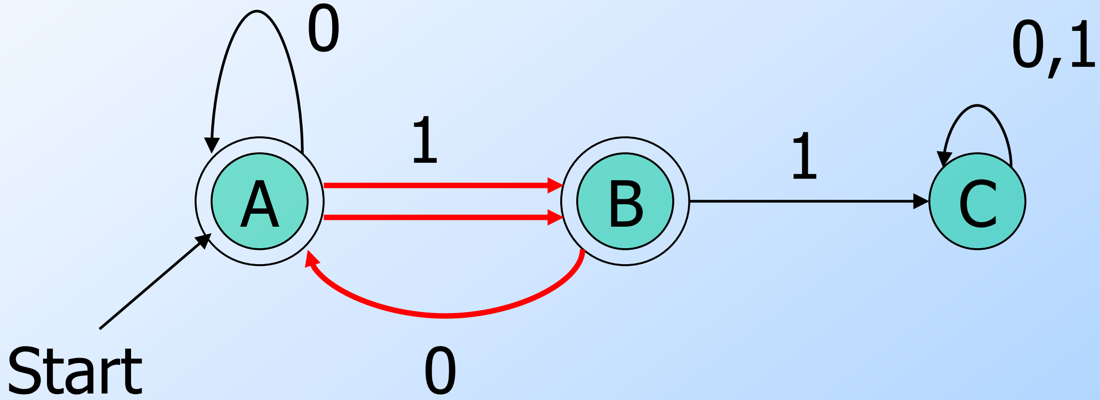
Then arc labeled 0 from current state B.



Example: String in a Language

String 101 is in the language of the DFA below.

Finally arc labeled 1 from current state A. Result is an accepting state, so 101 is in the language.



Example – Concluded

- The language of our example DFA is:

{w | w is in {0,1}* and w does not have two consecutive 1's}

Such that…

These conditions about w are true.

Read a *set former* as "The set of strings w…

Try Yourself

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

a) The set of all strings ending in 00

b) The set of all strings with three consecutive 0's (not necessarily at the end)

c) The set of strings with 011 as a substring

Try Yourself

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

a) The set of all strings such that each block of five consecutive symbols contains at least two 0’s

b) The set of al strings whose tenth symbol from right end is a 1

c) The set of strings such that the number of 0’s is divisible by five, and the number of 1’s is divisible by 3